UNIVERSIDAD POLITÉCNICA DE VALENCIA

TESINA DE MÁSTER

Diagnóstico distribuido mediante el Método de Anidamiento Latente de Fallos usando RdPC

Autor:
Miguel Sanjuan Verdu

Director:
Antonio Correcher Salvador

Septiembre 2011

Resumen

Hoy en día, los sistemas industriales son cada vez más grandes y complejos que resulta muy complicado realizar el control, la supervisión o el diagnóstico de forma centralizada, de ahí la idea de emplear sistemas distribuidos. En esta tesina se presenta una ampliación del método de Anidamiento Latente de Fallos usando Redes de Petri Coloreadas (Garcia *et al.*, 2008*a*)(Garcia *et al.*, 2008*b*) para su uso en SED's complejos distribuidos. En este método se tiene en cuenta un diagnosticador para cada subsistema y la interacción entre éstos sin afectar al sistema de control. Además se toma en consideración el flujo compartido de información y/o material que se produce en estos sistemas incluyendo el retardo en la propagación y recuperación de los fallos. Mediante esta metodología se obtiene un diagnóstico más preciso del fallo producido. También se presenta el caso práctico de la fabricación y paletizado automático de sacos de arena para poder comprender mejor el método presentado, utilizando un sistema Scada que realiza las labores de monitorización y control remoto, además del diagnóstico.

Índice general

Índice de figuras

Capítulo 1

Introducción y Organización del documento

1.1. Introducción

En la actualidad, el mantenimiento industrial es un área muy importante en cuanto a costos se refiere, donde se requieren sistemas de diagnóstico de fallos para obtener un plan de mantenimiento fiable y así conseguir una larga vida útil de la máquina tanto en productividad, como en efectividad y disponibilidad, y con ello, reducir así los costes de producción.

Un oportuno aislamiento y detección de fallo permite minimizar los riesgos humanos y daños al equipo durante el funcionamiento del sistema. Ésto es un requerimiento básico para la implementación en las estrategias de mantenimiento y sistemas de producción.

El objetivo del diagnóstico es determinar el estado de una planta física a través de las medidas de los sensores de la planta, el conocimiento previo de la planta y su comportamiento.

En la última década, la monitorización, detección de fallos y metodologías de diagnóstico basadas en DES se han utilizado en una amplia variedad de diferentes sistemas tecnológicos, desde sistemas de procesos hasta sistemas de transporte inteligentes.

Uno de los métodos más utilizados en el modelado de fallos para sistemas complejos es la utilización de técnicas de sistemas de eventos discretos (SEDs) (Lin, 1994). Dentro de éstos sistemas se encuentran las Redes de Petri (Murata, 1989), muy eficientes en el modelado de estos sistemas ya que proporcionan sincronismo, concurrencia, exclusión mutua y compartición de recursos, características que han aportado una mayor capacidad y potencia de representación en los modelos resultantes. También tienen la capacidad de aplicar técnicas de fusionado de lugares, permitiendo así reducir el tamaño de los modelos. Esta capacidad se acentúa más con las denominadas Redes de Petri Coloreadas (RdPC) (Jensen, 1995), que contribuyen a la aplicación de técnicas para el fusionado y poder representar distintos subprocesos concurrentes que coexisten en la misma estructura gráfica de la RdP. Las RdPC

también permiten que en sus arcos se asignen funciones con capacidad de transformaciones lineales.

Todas estas características de las RdP añadidas a las técnicas de fusión de las RdPC le dan la suficiente robustez al sistema para aplicar el diagnóstico de fallos a sistemas complejos.

En lo referente a modelos de diagnóstico, la mayor parte de ellos son sistemas centralizados (Garcia *et al.*, 2002; Hashtrudi Zad *et al.*, 2003), pero en muchas aplicaciones los sistemas son demasiado grandes y complejos para ser tratados como uno solo, de ahí la idea del procesamiento "por partes", y la idea de una arquitectura de supervisión distribuida. (Fabre *et al.*, 2002)

Un diagnosticador centralizado almacena el modelo entero de la planta, recibe la lectura de todos los sensores y ejecuta el algoritmo de diagnóstico, con los inconvenientes de la alta complejidad, la poca robustez y la pobre adaptabilidad. (Su *et al.*, 2002)

En cambio, los sistemas distribuidos aparecen en la industria de forma natural y se han aplicado a diferentes áreas como sistemas automáticos, redes de comunicación, calefacción, ventilación, AA, sistemas de transporte inteligentes y sistemas de procesos. (Genc and Lafortune, 2003)

En algunos casos se demuestra que con el diagnóstico distribuido se obtienen los mismos resultados que con un diagnóstico centralizado. (Aramburo-Lizarraga *et al.*, 2005)

En esta tesina se pretende aplicar una técnica de diagnóstico distribuido utilizando el método de Anidamiento Latente de Fallos. Al tratarse de una herramienta para el análisis de fallos en sistemas complejos centralizados, el objetivo es ampliar ésta para su uso en sistemas complejos distribuidos.

1.2. Organización del documento

La tesina está organizada en 9 capítulos organizados de la siguiente manera:

Capítulo 2: Se va a realizar una introducción a las Redes de Petri, así como una pequeña clasificación, destacando especialmente las Redes de Petri Coloreadas debido al uso que tienen en esta tesina.

Capítulo 3: En este capítulo se hace una introducción al diagnóstico de fallos así como a diferentes métodos de diagnóstico, realizando también una introducción al diagnóstico de fallos en sistemas de eventos discretos.

Capítulo 4: Se da una introducción a los sistemas de diagnóstico distribuido, describiendo algunas de las técnicas más utilizadas, como el diagnóstico modular centralizado, el diagnóstico distribuido para sistemas cualitativos y el diagnóstico distribuido usando RdP's.

Capítulo 5: En este apartado se hace la definición del método de Anidamiento Latente de Fallos, definiendo las etapas de diseño, definición del conjunto de fallos, lugares de anidamiento de fallo, definición de transiciones,...

Capítulo 6: Este capítulo describe la técnica de Anidamiento Latende de Fallos Distribuida (ALfD).

Capítulo 7: Se presenta una ampliación de la técnica de Anidamiento Latende de Fallos Distribuida, en la que se tiene en cuenta el retardo en la propagación y recuperación del fallo.

Capítulo 8: En este capítulo se presenta el caso práctico del método aplicado a la fabricación y paletizado de sacos de arena.

Capítulo 9: Por último se presentan las conclusiones del método de Anidamiento Latente de Fallos Distribuido y del método de Anidamiento Latente de Fallos Distribuido con retardo, así como posibles líneas de investigación para trabajos futuros.

Capítulo 2

Conceptos básicos de Redes de Petri

2.1. Redes de Petri (RdP's)

Carl Adem Petri introdujo estas redes en su tesis doctoral (Petri, 1962). Estas redes son una herramienta muy potente para su utilización en sistemas distribuidos ya que con ella se pueden modelar sistemas con capacidad de concurrencia, sincronización, exclusión mutua y conflictos.

Durante todos estos años se han realizado diferentes trabajos con RdPs, dando lugar a una gran variedad de las mismas. Las RdPs más destacadas son las estocásticas (Florin *et al.*, 1991), temporizadas (Berthomieu and Diaz, 1991), algebraicas (Kan and He, 1996), continuas e híbridas (David and Alla, 2005) y difusas (Pedrycz and Gomide, 1994), otra que ha tenido mucha relevancia son las Redes de Petri Coloreadas, en las que se entra un poco más en detalle en la Sección 2.2.

Como se puede ver en la Figura 2.1, una RdP se representa de forma gráfica como un gráfico compuesto por unos elementos denominados lugares, transiciones y arcos. Los lugares se representan mediante círculos y en ellos se encuentran las marcas, representadas por puntos, las transiciones son segmentos de recta que se asocian a eventos, y los arcos son segmentos de recta que unen los lugares con las transiciones y viceversa.

Definición 1 *Una Red de Petri ordinaria se define como una cuádrupla:*

$$N = \{P, T, Pre, Post\} \tag{2.1}$$

donde: $P = \{p_1, p_2, ..., p_m\}$ *es un conjunto finito de lugares;*
$T = \{t_1, t_2, ..., t_n\}$ *es un conjunto finito de transiciones;*
$P \cap T = 0$ *y* $P \cup T \neq 0$*;*
$Pre : PxT \rightarrow \{0, 1\}$ *relaciona los lugares con las transiciones;*
$Post : TxP \rightarrow \{0, 1\}$ *relaciona las transiciones con los lugares;*

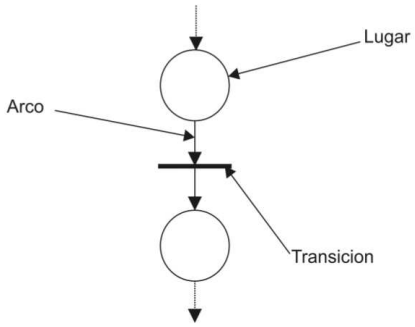


Figura 2.1: Lugar, transición y arco en una Red de Petri

El concepto de marcado hace referencia a un vector de marcas que se ubican en los lugares y representan el estado en el que se encuentra el proceso. *M* es el marcado de una red, que se define como un vector columna de *n* marcas, es decir, $M = (m_1, m_2, ..., m_n)$.

Una Red de Petri marcada sería un par $N_m = \{N, m_0\}$, en el cual *N* es una Red de Petri ordinaria y m_0 es el marcado inicial. El disparo de las transiciones provoca la evolución del marcado.

Una RdP se define con dos matrices, $n = |P|$ (número de lugares de P) y $m = |T|$ (número de transiciones de T).

Se denomina:

Matriz de incidencia previa a la matriz $C^- = [C\overline{ij}]_{nxm}$ en la que $c_{\overline{ij}} = Pre(p_i, t_j)$

Matriz de incidencia posterior a la matriz $C^+ = [C\overset{+}{ij}]_{nxm}$ en la que $\overset{+}{c_{ij}} = Post(t_j, p_i)$

Matriz de incidencia de $N : C = C^+ - C^-$

Las RdP tienen diferentes tipos de nodos, los nodos OR y los AND. Un nodo OR es cuando un lugar que posee varios arcos de entrada y/o salida, presentando dos casos particulares:

Nodo de selección: Un único arco de entrada y dos o mas de salida.
Nodo de atribución: Un único arco de salida y dos o mas de entrada.

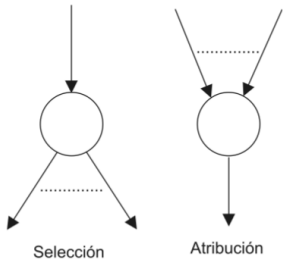


Figura 2.2: Nodos tipo OR

Los nodos AND es cuando una transición tiene varios arcos de entrada y/o salida, con dos casos particulares:

Nodo de distribución: Un único arco de entrada y dos o mas de salida.
Nodo de conjunción: Un único arco de salida y dos o mas de entrada.

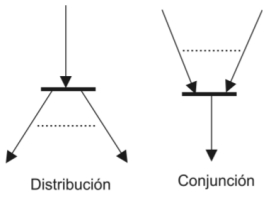


Figura 2.3: Nodos tipo AND

Una transición se llama disparable o validada si cada uno de los lugares de entrada a ésta contiene al menos una marca. El disparo solo se puede realizar cuando la transición esté validada y acontezca un evento asociado a ésta. El disparo consiste en quitar las marcas de los lugares de entrada a una transición y poner una marca a los diferentes lugares de salida de ésta.

Si se desea conocer el marcado alcanzable de una RdP, solo hay que resolver la siguiente ecuación:

$$M_f = M_i + W\underline{S} \tag{2.2}$$

donde:
$M_f =$ marcado final;
$M_i =$ marcado inicial;
$W =$ la matriz de incidencia $= C^+ - C^-$;
$\underline{S} =$ secuencia de disparo.

2.1.1. Clasificación y tipos de RdPs

En este apartado se presentan varias extensiones de las Redes de Petri, realizando una pequeña explicación de cada una.

Red de Petri viva

El concepto de vivacidad indica que la RdP no quedará bloqueada, esto quiere decir que no presenta ninguna transición que no pueda ser disparada a partir de un cierto instante. Una

explicación más formal sería: una transición t_j es viva para un marcado inicial m_0, si para cada marcado alcanzable m_i existe una secuencia de disparo S desde m_i, que contenga a m_j.

Red de Petri limitada

Una Red de Petri es limitada si cada lugar tiene un número máximo de marcas que puede contener. Una definición formal sería: Una RdP es limitada para un marcado inicial m_0, si todos los lugares son limitados para m_0.

Este tipo de redes presentan un caso especial, en el caso de que únicamente posean una marca, reciben el nombre de RdP binarias, y de ellas se deriva el Grafcet (Garcia, 2000), muy utilizados en los Autómatas Programables.

Redes de Petri no-autónomas

Estas RdP se caracterizan por que su evolución es condicionada por eventos externos al sistema, donde cada transición se asocia a un evento, activando esta transición cuando el evento ocurre. Los eventos corresponden a cambios en las lecturas del sistema.

Redes de Petri sincronizadas

Se denotan como $\{R, E, Sync\}$ donde R es una RdP marcada, E es un conjunto de eventos y $Sync$ es una función de transiciones.

Redes de Petri temporizadas

El tiempo influye en los procesos industriales en aspectos como: tiempo de fabricación, índice de producción, capacidad, flexibilidad, etc. Esto provoca que existan dos tipos de redes temporizadas, las P-temporizadas y las T-temporizadas, refiriéndose a los lugares y a las transiciones respectivamente.

> ***Redes de Petri P-temporizadas*** Es un par $< R, Tempo >$ donde, R es una RdP marcada y $Tempo$ es una función del conjunto de P lugares. La notación $Tempo(p_i) = d_i$, se refiere a la temporización asociada a un lugar p_i.
>
> ***Redes de Petri T-temporizadas*** La transición está asociada a la temporización t_i correspondiente.

Redes de Petri interpretadas

Las Redes de Petri Interpretadas (RdPI) permiten asociar señales de entrada y salida a los modelos de RdP. Se define como $R = \{N, m_0\}$ donde N es una RdP ordinaria y m_0 el marcado para esa RdP.

Redes de Petri estocásticas

A estas Redes de Petri se les asocia a cada transición una variable aleatoria con una distribución exponencial para expresar el retardo de habilitación hasta el disparo de la transición. Estas redes se suelen utilizar en protocolos de comunicación. Se define como $E = \{R, L\}$ donde R es una RdP marcada y $L = (l_1, l_2, ..., l_m)$ es el conjunto de retardos.

Redes de Petri continuas

En algunos casos un sistema discreto se puede volver muy complicado, debido al gran número de estados alcanzables, una forma de solucionarlo es continuizar el sistema, lo que permite utilizar herramientas de programación lineal o ecuaciones diferenciales para simplificar el problema. No todas las RdP discretas se pueden continuizar y las propiedades de una a otra pueden variar, esto quiere decir que si un sistema discreto está libre de bloqueos no significa que el continuo también lo esté.

Redes de Petri híbridas

Muchos procesos pueden estar controlados a la vez por variables discretas y continuas, de ahí la necesidad de disponer de una RdP que pueda contener lugares y transiciones continuas, así como lugares y transiciones discretas. Un marcado discreto puede convertirse a su vez en un marcado continuo y viceversa.

2.2. Redes de Petri Coloreadas (RdPC's)

Kurt Jensen en 1979 formuló las Redes de Petri Coloreadas, siendo éstas una evolución de las Redes de Petri (Petri, 1962). Estas redes son un lenguaje gráfico orientado de propósito general, útil para especificar, diseñar y analizar sistemas concurrentes.

Las RdP se definen matemáticamente, ofreciendo la ventaja de poder ejecutarlas en cualquier herramienta de programación computacional.

La sintáctica de un RdPC se define igual que una RdP general, lugares, transiciones y arcos, con la modificación de que a ésta se le añaden las inscripciones, que son de tipo textual mientras que el resto son de tipo gráfico.

Los lugares se representan con elipses, las transiciones mediante rectángulos y los arcos por segmentos de recta que unen los lugares con las transiciones y viceversa. Las inscripciones de arco, son funciones y se usan para determinar la cantidad de marcas que se mueven entre los lugares (Figura 2.4).

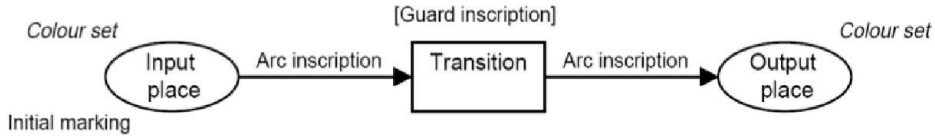


Figura 2.4: Elementos de una Red de Petri Coloreada

Definición 2 *Una Red de Petri Coloreada se define como una séxtupla:*

$$C = \{P, T, Pre, Post, m_0, C\} \tag{2.3}$$

Donde P, T, Pre, $Post$, m_0 se definen igual que para una RdP generalizada, mientras que C representa el conjunto de colores $C = (C_1, C_2, ..., C_n)$.

De forma más formal se define $Pre(P_i, T_j/C_k)$ y $Post(T_j, P_i/C_k)$, correspondiendo en el caso general a la combinación lineal de las marcas coloreadas relacionadas al lugar P_i.

En la Figura 2.5 se muestra un pequeño ejemplo para que se pueda entender mejor la definición de RdPC, en el que se puede ver que el lugar está representado por un círculo, pudiendo contener varias marcas de un mismo o varios colores, la transición se representa por una línea a la que se le asocia un conjunto de colores disparados, cada uno representando una posibilidad diferente de disparo, y un arco que conecta un lugar con una transición y viceversa. El peso de un arco es la función Pre o $Post$, la cual establece una correspondencia entre cada color de las transiciones y cada color del lugar, según la dirección del arco.

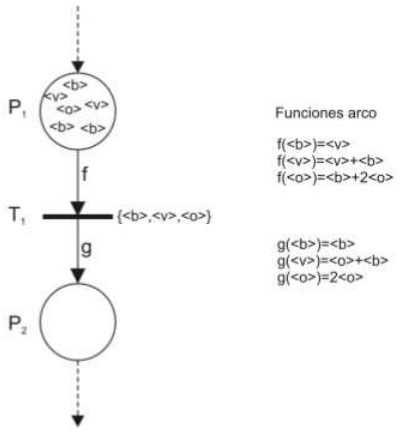


Figura 2.5: Parte de una RdPC

Una transición está habilitada con respecto al color C_k para un marcado M, si y solo si el número de marcas contenidas en todos los lugares de entrada P_i de T_j son mayores o igual a $Pre(P_i, T_j/C_k)$.

$$M(P_i) \geq Pre(P_i, T_j/C_k), \forall P_i \in T_j \tag{2.4}$$

Donde $Pre(P_i, T_j/C_k)$ es la imagen del color C_k por el peso de la función del arco que conecta el lugar P_i con la transición T_j.

Un disparo T_j/C_k puede añadir o quitar marcas de un lugar a otro, provocando operaciones como las siguientes:

- Se quitan de todos los lugares P_i de T_j si hay un número de marcas igual o mayor a $Pre(P_i, T_j/C_k)$.
- Se añaden a todos los lugares de salida P_i de T_j si hay un número de marcas igual o mayor a $Post(T_j, P_i/C_k)$.

Obteniendo como resultado del disparo de T_j con respecto al color C_k un nuevo marcado M'.

$$M'(P_i) = M(P_i) + Post(T_j, P_i/C_k) - Pre(P_i, T_j/C_k), \forall P_i \tag{2.5}$$

En el caso de tener una secuencia de disparos S, por ejemplo $S = T_1/C_{h1} \cdot T_2/C_{h2} \cdot \ldots \cdot T_k/C_{hk}$, entonces la expresión para el nuevo marcado M_{k+1} es:

$$M_{k+1}(P_i) = M_1(P_i) + \sum_{j=1}^{k} Post(T_j, P_i/C_k) - Pre(P_i, T_j/C_k) \tag{2.6}$$

Las Redes de Petri Coloreadas tienen un gran uso en esta tesina ya que el diagnóstico se realiza utilizando estas redes, dándole las ventajas de concurrencia, sincronismo, recursos compartidos y simplificación de modelos entre otras muchas ventajas.

Capítulo 3

Conceptos básicos del Diagnóstico de Fallos

3.1. Definición formal y fundamentos

En este capítulo se va a realizar una introducción resumida de algunos de los fundamentos básicos de las técnicas de diagnóstico. También se van a nombrar algunas características destacables, clasificación y unos de los métodos más comunes.

Definición 3 *El diagnóstico de fallos es una monitorización de un sistema que es usado para detectar fallos y diagnosticar su localización y significancia en el mismo sistema (Chen and Patton, 1999).*

Los fallos se pueden definir como un comportamiento anormal del sistema, pudiéndose distinguir los siguientes tipos: fallos de procesos de tipo aditivo, fallos de proceso de tipo multiplicativo, fallos de sensores y fallos de actuadores (Gertler, 1998).

Las tareas necesarias para realizar el análisis y diagnóstico de fallos son:

- *Detección del fallo:* Indica si ha ocurrido o no un fallo.

- *Aislamiento del fallo:* Localiza el fallo.

- *Identificación del fallo:* Determina la naturaleza del fallo.

En la mayoría de sistemas de diagnóstico solamente se realizan las fases de detección y aislamiento.

3.2. Métodos de diagnóstico

Debido a la variedad de campos en los que se utiliza el diagnóstico de fallos, han surgido diferentes métodos para la detección de los mismos. Principalmente se pueden dividir en tres grupos: Métodos basados en la historia del proceso, métodos basados en el conocimiento y la combinación de ambos.

Los métodos basados en la historia del proceso utilizan técnicas de clasificación, reconocimiento de patrones y técnicas estadísticas. Los métodos basados en el conocimiento necesitan un conocimiento más detallado del proceso, siendo éstos métodos difíciles de implementar y mantener.

En la Figura 3.1 se puede observar una posible clasificación de los algoritmos de diagnóstico más comunes y utilizados.

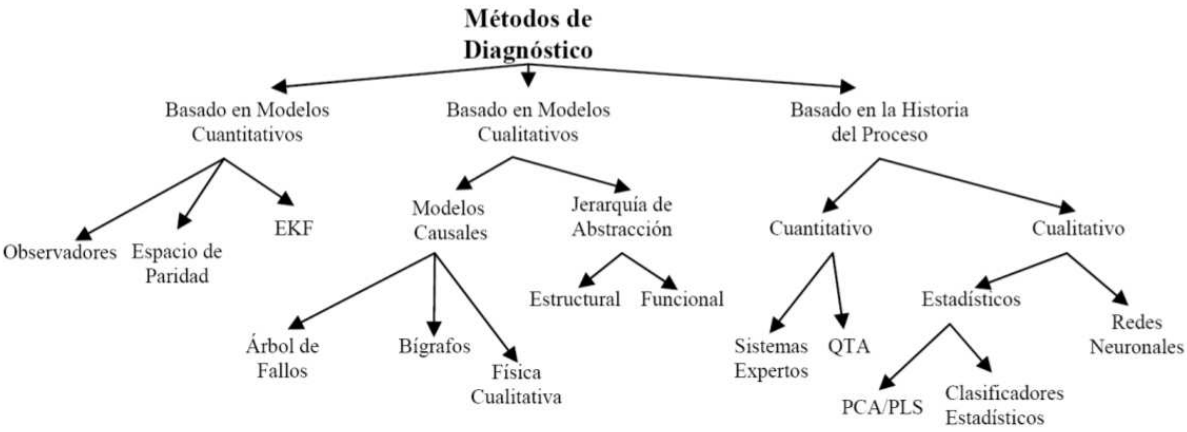


Figura 3.1: Clasificación de algoritmos de diagnóstico

La evolución de los sistemas conlleva que éstos se hayan vuelto más robustos y complejos, lo que ha provocado que se pase de tratar el diagnóstico como un conjunto total a desglosarlo en subconjuntos para así poder tratar cualquier tipo de sistema de forma más sencilla.

A continuación se describen brevemente algunos de los métodos más utilizados:

Métodos basados en conocimiento

Las técnicas que utilizan estos métodos se basan en un conocimiento detallado del proceso para realizar el diagnóstico. Dentro de éstos están los modelos analíticos y los cualitativos.

Métodos Cuantitativos

Los métodos cuantitativos emplean el diagnóstico de fallos basado en modelos, la forma más comúnmente utilizada para realizar este diagnóstico de fallos es mediante el análisis de residuos.

Hay dos pasos principales: la generación de residuos y la identificación de la causa.

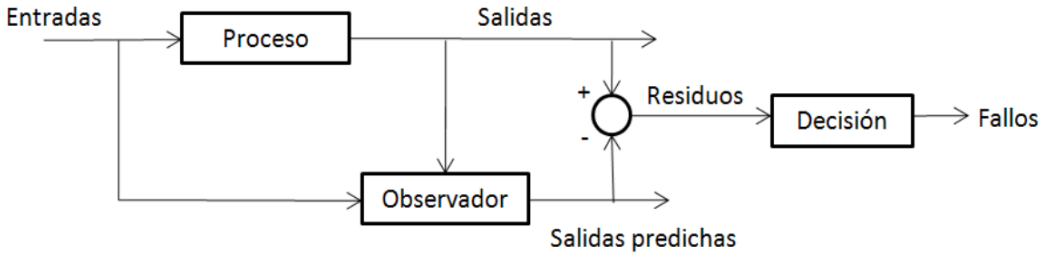


Figura 3.2: Esquema general basado en método de observadores

Los principales métodos de esta clasificación son los basados en observadores, relación de paridad y estimación de parámetros.

En la Figura 3.2 se muestra un esquema general clásico del método basado en observadores donde la estimación de salida del error es usada como un residuo.

El método de la relación de paridad consiste en analizar las medidas del sistema, tanto las entradas como las salidas. El valor residuo no siempre deberá ser cero, debido a perturbaciones en el sistema tales como el ruido, imprecisiones, etc..

El método basado en estimación de parámetros utiliza ecuaciones diferenciales parciales y ordinarias, ofreciendo modelos dinámicos precisos del sistema pero con un alto coste computacional.

Métodos Cualitativos

Partiendo de que no es posible obtener un modelo matemático exacto del sistema, se crean otros métodos capaces de manejar información incompleta del sistema.

Una de las metodologías muy empleadas son los algoritmos basados en sistemas expertos. Esta técnica intenta imitar el conocimiento de un ser humano para resolver un problema.

En los sistemas expertos se utilizan una serie de reglas del tipo *IF-THEN-ELSE* y un motor de inferencia para sacar conclusiones del comportamiento del proceso. Un sistema experto puede resultar fácil de realizar e implementar, aunque presenta desventajas frente a nuevas condiciones que no se hayan tenido en cuenta en la base de reglas.

Los grafos de señal (*Signed Directed Graphs*, SDG) son una herramienta gráfica que representa las variables de proceso como nodos gráficos, que representan un evento o variable, y las relaciones causales por arcos, que unen los diferentes nodos.

Una de las técnicas basadas en grafos es la denominada árbol de fallos, siendo una herramienta deductiva de análisis de fallos utilizada y perfeccionada continuamente en diversas industrias. Este método parte de la selección de un "suceso no deseado o evento que se pretende evitar", y así conseguir averiguar los orígenes del mismo.

El árbol de fallos está compuesto por nodos, distribuidos en niveles superiores o inferiores, representando el fallo de cabecera y los subfallos, dando ese aspecto de árbol. Los niveles se conectan con puertas lógicas del tipo *AND* u *OR* y se desarrolla hasta que se llegua a unos "sucesos básicos", es decir, que no precisan de otros fallos para ser explicados.

Aunque parezca que se trata de un método anticuado, ya que lleva muchos años utilizándose y puede llevar varios años confeccionar una estructura de árbol de forma robusta, se sigue utilizando por su sencillez y robustez en procesos complejos.

Métodos basados en la historia del proceso

Estos métodos solamente necesitan una cantidad de registros de históricos del sistema para realizar el diagnóstico.

Uno de los métodos más utilizados para el diagnóstico con estos métodos, son los sistemas expertos. También son muy conocidas las Redes Neuronales, que son del tipo no estadístico, mientras que el Análisis de Componentes Principales (PCA) y Mínimos Cuadrados Parciales (PLS) son estadísticos.

Las Redes Neuronales tienen muy buenas propiedades para el diagnóstico de fallos, ya que aprenden a diagnosticar fallos gracias al entrenamiento de datos, siendo también tolerantes al ruido y tener buena capacidad de adaptación on-line.

El Análisis de Componentes Principales (PCA),es un análisis multivariable y se utiliza en procesos de gran cantidad de información. Esta característica se debe a que permite reducir la dimensión del modelo con el uso de dependencias lineales entre las diferentes variables.

También se utiliza mucho el Análisis de Tendencias Cualitativas (QTA), no solamente se usa para el diagnóstico de fallos sino también para el seguimiento de estado de las variables más importantes para de esta forma poder prevenir posibles fallos futuros.

Métodos basados en combinaciones

Algunos métodos tales como los métodos Neuro-Fuzzy (NF) se pueden fusionar para así obtener una mayor robustez, en este caso para combinar la capacidad de reconocimiento de patrones y adaptabilidad con la capacidad de inferencia.

3.3. Diagnóstico de Fallos en Sistemas de Eventos Discretos (SED's)

En esta sección se va a hablar de los principales métodos usados en el diagnóstico de fallos en sistemas discretos, ya que en la sección anterior se ha hablado de los métodos usados para sistemas continuos.

La mayoría de procesos industriales actuales son de naturaleza híbrida, es decir, combinan variables de tipo continuo y discreto en el mismo sistema.

Las técnicas de diagnóstico en sistemas SEDs utilizan Lenguajes Regulares, Grafos de Estado, Máquinas de Estado Finito (MEFs) y Redes de Petri (RdPs), siendo estas últimas muy difundidas en la actualidad.

Principalmente se han venido utilizando MEFs (Sampath *et al.*, 1995) para el diagnóstico en sistemas SEDs, pero en trabajos actuales se está enfocando al uso de RdPs debido a su potencia matemática y formalismo gráfico.

A partir de los modelos de eventos discretos se construyen diagnosticadores como por ejemplo (Ramadge and Wonham, 1987; Viswanadham and Johnson, 1988; Genc and Lafortune, 2003), que transforman todos los modelos individuales en un diagnosticador global, capaz de diagnosticar múltiples fallos. El inconveniente que presenta esta técnica es que su complejidad aumenta al aplicarse a procesos complejos de mediana y alta escala.

Una solución al problema anterior se plantea en (Moreno, 2000), descomponiendo los procesos en subsistemas, generando así diagnosticadores más sencillos y fáciles de tratar. Otra solución son los diagramas de decisión binaria ordenada (OBDD), que codifica los estados del sistema en funciones booleanas y las representa como pesos en una Red de Petri limitada (Xue and Yan, 2007).

Los problemas de diagnóstico se pueden tratar de forma off-line u on-line. El diagnóstico off-line realiza una serie de pruebas al sistema y observa el comportamiento de las salidas, comprobando que se ajusten a lo definido como comportamiento normal. El diagnóstico on-line se realiza mientras el sistema se encuentra en funcionamiento, comprobando la secuencia de funcionamiento normal y teniendo en cuenta sucesos inesperados que puedan ocurrir en cualquier momento.

Algunas aplicaciones más recientes utilizan técnicas fuzzy (Kiliç *et al.*, 2006), construyendo una base de reglas a partir de los síntomas o eventos de fallo, en donde cada evento fuzzy tiene una prioridad diferente, obtenida con el algoritmo de clustering K-means, presentando el inconveniente de que es necesario un gran número de datos para realizar el clustering y elevando así el coste computacional.

Las Redes de Petri Coloreadas se pueden utilizar para el control y diagnóstico de fallos, mostrando en (Kasturia *et al.*, 1988) un ejemplo de la capacidad de análisis y modelado.

También se utilizan las Redes de Petri Coloreadas Temporizadas (RdPCT), así como el modelo del control estadístico del proceso (SPC).

Otra técnica también muy reciente y que se va a utilizar en esta investigación es el Anidamiento Latente de Fallos (ALF, o LNM en sus siglas en inglés) (Rodríguez, 2009), la cual se explica más en profundidad en el Capítulo 5.

Capítulo 4

Diagnóstico modular

Existen principalmente dos métodos para el diagnóstico de fallos en sistemas distribuidos. El primero de ellos consiste en la utilización de un diagnosticador local para cada subsistema que se comunica con un proceso de coordinación que actúa como una unidad centralizada (Garcia *et al.*, 2008*b*; Rodríguez *et al.*, 2008*a*,*b*, 2010; Correcher *et al.*, 2001). El segundo consiste en un diagnosticador local para cada subsistema que se comunica con los otros diagnosticadores (Fabre *et al.*, 2002; Genc and Lafortune, 2003; Aramburo-Lizarraga *et al.*, 2005). Dentro de éstos dos métodos se encuentran diferentes mejoras/modificaciones de los mismos. A continuación se describen brevemente algunas de las técnicas mas utilizadas.

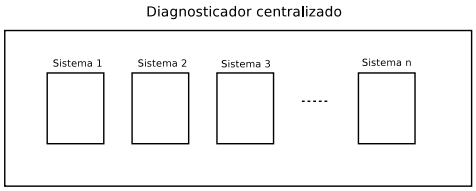


Figura 4.1: Diagnóstico centralizado

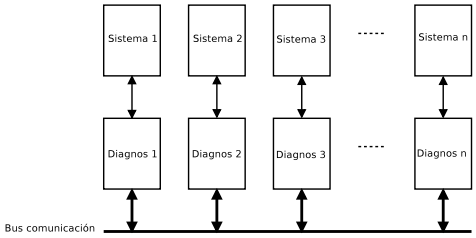


Figura 4.2: Diagnóstico distribuido

4.1. Técnicas de Diagnóstico Modular

4.1.1. Diagnóstico Modular Centralizado

Este método se presenta en (Garcia *et al.*, 2002), y estudia la descomposición modular como una aproximación al diagnóstico de fallos basado en DES, analizando el problema de acoplamiento producido por la implementación de un diagnosticador modular centralizado.

Una de las ventajas de este método es que no requiere de un modelo detallado del sistema para realizar el diagnóstico. Por ese motivo, este método se puede adaptar a grandes y complejos sistemas, no siendo necesariamente DES, sino que también se puede aplicar a sistemas dinámicos continuos y modelarlos como ecuaciones diferenciales.

El método asume que hay dos niveles en el sistema. Un nivel bajo que actúa como controlador local y un nivel alto que actúa como supervisor/monitorización, llevando a cabo las labores de coordinación de las tareas de control así como las tareas de diagnóstico y recuperación de fallos.

La adecuada implementación de métodos de diagnóstico de fallos es necesaria en el nivel de monitorización.

La interfaz entre los dos niveles transmite información entre los supervisores cuando se activa algún evento del supervisor.

En los sistemas DES, el modelo del sistema es un modelo exhaustivo que incorpora no solo el comportamiento normal de varios componentes, sino también estados adicionales representados por fallos de éstos. La fase de modelado empieza con el modelado de cada componente individualmente que componen el sistema.

El modelo del sistema está formado por la composición de todos los modelos individuales. Cada componente individual se modela por la siguiente expresión general,

$$G_i = (X_i, \Sigma_i, \delta_i, x_{0i}) \tag{4.1}$$

donde X_i es el espacio de estados, Σ_i son los eventos de activación, δ_i es la función de transición, y x_{0i} es el estado inicial de G_i. Algunos eventos en δ_i son observables y otros no observables, $\Sigma_i = \Sigma_o \cup \Sigma_{uo}$. Los eventos no observables son los que no dependen directamente de las medidas de los sensores o estados, en éstos eventos se incluyen los fallos del sistema.

El inconveniente del diagnóstico de fallos basado en DES es que presenta un crecimiento exponencial del número de estados al incorporar nuevos fallos o componentes, teniendo en cuenta que la complejidad del sistema se puede considerar dependiendo del número de dispositivos relacionados entre sí, dando lugar a la descomposición modular para resolver estos problemas.

La descomposición modular puede crear fallos de diagnóstico debidos al fenómeno de acoplamiento, este fenómeno de acoplamiento se produce cuando los subsistemas comparten algún tipo de flujo de energía o material.

La consecuencia del flujo compartido, desde el punto de vista del diagnóstico, es que un subsistema que funciona con normalidad puede alcanzar un estado de fallo, debido a que no recibe flujo de otro subsistema.

Para dividir el sistema en subprocesos, se deben tener en cuenta las siguientes propuestas:

a) Los subprocesos se formarán teniendo en cuenta la relación entre los componentes que forman el sistema, especialmente la cooperación entre los subsistemas.

b) Tener en consideración las dimensiones de los modelos resultantes.

La descomposición modular da lugar a la descomposición del sistema en subsistemas, cada uno con su correspondiente controlador que envía información a un diagnosticador modular centralizado, asumiendo que cada diagnosticador modular diagnostica a su correspondiente subsistema.

El modelo global de esta descomposición se podría representar con la media de los diferentes subsistemas.

$$G_M = G_{S1} \parallel ... \parallel G_{Sn} \tag{4.2}$$

Cada subsistema estará formado por los modelos de los *m* componentes locales del sistema S_i.

Obteniendo un modelo global como el siguiente.

$$G_M = (Q_M, \Sigma_M, \delta_M, q_{0M}) \tag{4.3}$$

Es fácil comprobar que el número de estados necesarios para empezar el diseño de la descomposición modular se ha reducido significativamente en comparación con el caso integral.

Ésta metodología para el diagnóstico basada en FSM se puede aplicar también utilizando RdP. Al utilizar las RdP, se podría obtener una optimización del tamaño del controlador global, aplicando técnicas de sincronización de transiciones y fusión de lugares equivalentes.

El comportamiento del diagnosticador está condicionado a la existencia de lazos, determinados por la función de transición δ_{dSi}.

El fenómeno de acoplamiento en un diagnosticador modular G_{dSi} tiene lugar cuando ocurre un fallo de acoplamiento $F_c \in \Sigma_{Sj}$ en un subsistema S_j, que comparte flujo de materiales o energia con el subsistema S_i.

El acoplamiento no es deseable, ya que éste puede producir el bloqueo del sistema. También, cuando un fallo permite el funcionamiento de manera degradada del sistema, el resto de diagnosticadores deben ser capaces de detectar la ocurrencia de un fallo adicional.

La solución del problema de acoplamiento, está precedida por un análisis de los posibles acoplamientos.

Se debe utilizar información no local para detectar el fallo en situaciones susceptibles a acoplamiento, es decir, utilizar información de otros diagnosticadores en los que ha habido algún fallo y así informar al resto de ello.

Los eventos serán por tanto clasificados en locales y no locales $\Sigma_{Si} = \Sigma_{LSi} \cup \Sigma_{NLSi}$.

Esta función se puede construir incluyendo información de desacoplo I_{dSj}, la cual notificará que ha ocurrido un fallo en otro diagnosticador y proporcionará una función para el diagnosticador acoplado.

El método presentado tiene las ventajas de que ofrece un claro criterio para la descomposición modular, analizando el problema de acoplamiento y aportando soluciones al mismo, pero también presenta inconvenientes, tales como la complejidad del modelo al utilizar arquitectura basada en autómata, provocando que no se pueda utilizar en sistemas muy complejos y también el ser un sistema centralizado, puesto que está demostrado que un sistema distribuido funciona mejor. De ahí que no utilicemos este método para nuestro propósito, solamente tomamos la idea de la descomposición y el flujo compartido.

4.2. Técnicas de Diagnóstico Distribuido

4.2.1. Diagnóstico Distribuido para Sistemas Cualitativos

En este método (Su *et al.*, 2002), se propone un nuevo método de diagnóstico distribuido on-line, en donde cada componente tiene su propio diagnosticador local, construido en base al conocimiento del componente.

Cada diagnosticador local se conecta con otro diagnosticador local basándose en las relaciones entrada/salida de los componentes asociados. El diagnosticador global es la suma de los estados de todos los diagnosticadores locales, reduciendo así la complejidad del sistema.

De ésta forma es sencillo añadir, quitar o modificar componentes, ya que solamente afecta a los componentes con los que está relacionado el componente a modificar, ofreciendo una mayor escalabilidad.

Cada diagnosticador local se basa principalmente en su observación local, la comunicación con los demás diagnosticadores locales se realiza para el perfeccionamiento del diagnóstico, de forma que si un diagnosticador local o canal de comunicación falla, otro diagnosticador local puede producir el diagnóstico a través de un canal de comunicación en buen estado, dando al sistema una mayor robustez.

En el diagnosticador hay unas trayectorias predefinidas debido a que hay estados a los cuales el sistema no puede alcanzar debido a limitaciones físicas y serían detectadas como fallo.

Para evaluar el estado de fallo, primeramente hace una estimación del comportamiento del sistema y la compara con las lecturas.

Una vez estimado el fallo se realiza el proceso de comunicación, enviando un mensaje de fallo a todos los diagnosticadores.

La idea es que cada proceso de comunicación se divida según se realice la dirección de la comunicación, ya sea “aguas arriba” o “aguas abajo”. Cuando un diagnosticador local envía un mensaje a otros diagnosticadores, pone una etiqueta al mensaje que indica que ha llegado a un punto fijo o no. Esta etiqueta también puede servir para determinar el fin de la comunicación.

Es posible que no todos los posibles candidatos a fallos contengan el fallo, esto es debido a que hay ambigüedad en la detección, para determinar la ocurrencia del fallo se necesita conocer más información sobre el sistema.

Para solucionar el problema de ambigüedad se deben excluir soluciones imposibles en el diagnóstico, realizando en cada diagnosticador local un pre-procesamiento de fallo antes de realizar un diagnóstico más avanzado, siendo una forma de reducir el consumo de recursos.

Las observaciones del sistema se pueden usar de dos formas:

a) Realizando el diagnóstico utilizando todas las observaciones anteriores.

b) Realizando el diagnóstico utilizando solamente las observaciones más recientes.

El modelo de diagnóstico local se expresa como,

$$\mathbb{D}_i = (\mathbb{G}_i, E_i, R_i, C_i) \tag{4.4}$$

donde $\mathbb{G}_i$ es el modelo cualitativo del sistema, E_i es la estimación, R_i es el informe local de fallos y C_i es el protocolo de comunicación.

Hay un reloj global para la sincronización de los diferentes diagnosticadores locales. Cada diagnosticador local observa los eventos sucedidos en cada intervalo.

El procedimiento para el diagnóstico distribuido sería el siguiente:

1) **Inicialización:** Iniciar el diagnosticador con los valores estimados iniciales.

2) **Diagnóstico:** En cada intervalo de tiempo se comprueba si hay alguna modificación en la observación. Si no ha cambiado, se actualiza el estado al valor anterior, y si hay modificación,

   a.- Calcular la estimación del estado anterior.

   b.- Calcular el estado del diagnosticador. En este caso se realiza la comunicación con el resto de diagnosticadores.

En este método se demuestra realizando una comparación que el diagnóstico distribuido funciona mejor que el diagnóstico centralizado, ya que necesita menos tiempo de computación y genera un número mayor de modelos de diagnóstico.

El método presenta las ventajas de que reduce la complejidad del sistema, ofrece escalabilidad y robustez, así como un buen diagnóstico de fallos, también hace una comparación con un sistema centralizado, demostrando que el sistema distribuido funciona mejor, pero también presenta inconvenientes, tales como el utilizar arquitectura basada en autómata, así como el tener que realizar una pre-compilación en cada diagnosticador y buscar una distribución óptima. Debido a los inconvenientes se decide no utilizar este método para nuestro propósito.

4.2.2. Diagnóstico Distribuido usando RdP's

RdP's con lugares fronterizos

En (Genc and Lafortune, 2007) se estudia la detección de fallos on-line y el aislamiento de los mismos en sistemas dinámicos modulares modelados como redes de Petri con lugares fronterizos. Los lugares en común entre las redes de Petri modelan el acoplamiento existente entre varios sistemas.

Muchas técnicas de diagnóstico se basan en el uso de modelos de autómata, pero el uso de redes de Petri en vez de modelos de autómata ofrece ventajas en el modelado de sistemas y análisis, especialmente en sistemas distribuidos.

Los modelos de redes de Petri se han usado para resolver problemas de observabilidad, monitorización, alarmas y diagnóstico de fallos en diferentes trabajos, recibiendo cada vez más atención en el uso en sistemas distribuidos (Genc and Lafortune, 2003).

Cada diagnosticador tiene información sobre su modelo local, y observa los tipos de fallos de los diagnosticadores a los que está asociado mediante los lugares fronterizos.

En este caso se considera un sistema modular de M lugares fronterizos de la red de Petri, presentando dos nuevos algoritmos de comunicación, uno es un sistema mejorado de comunicación DDC-2 y otro un DDC-M con tamaño fijo de mensaje.

El sistema a diagnosticar es un conjunto $\mathscr{S}$, definido como:

$$\mathscr{S} = \{(\mathscr{M}_m, \mathscr{P}_m : m = 1, 2, ..., M\} \tag{4.5}$$

donde $\mathscr{M}_m$ es una red de Petri y $\mathscr{P}_m$ son los lugares en común.

Se asume que los lugares fronterizos operan como una sola entidad, es decir, hay un reloj global para coordinar las diferentes redes de Petri.

La ventaja que ofrece la modularidad es que al no compartir información estructural, los subsistemas se pueden aislar o reemplazar sin que los demás subsistemas se tengan que modificar por completo.

El estado del diagnosticador también lleva información de diagnóstico, que provee información del tipo de fallo que ha ocurrido. También guarda un histórico de los cambios en los lugares en común con cada diagnosticador vecino.

En general la comunicación de una red de Petri tiene tres partes; un conjunto de estados, etiquetas de fallo y etiquetas de mensajes para cada vecino.

Una vez definidos los fallos, el diagnosticador se define como el par $\mathscr{S}_{\mathscr{D}} = (\mathscr{D}_m, \mathscr{P}_m)$, donde $\mathscr{D}_m = (\mathscr{M}_m, \triangle_{f,m})$, siendo $\triangle_{f,m}$ el conjunto de tipos de fallos.

El estado del diagnosticador del modulo $\mathscr{D}_m$ es una matriz de la forma:

$$\left(\begin{array}{c|c|c} - & - & - \\ x_s^m(i) & x_f^m(i) & x_l^m(i) \\ - & - & - \end{array}\right) \tag{4.6}$$

donde $x_s^m(i)$ es el estado del diagnosticador, $x_f^m(i)$ denota la etiqueta de fallo y $x_l^m(i)$ denota la etiqueta del mensaje.

Las etiquetas de fallo se utilizan al igual que en el modelo autómata para memorizar la ocurrencia de un evento de fallo.

Si los diagnosticadores no son informados añadiendo/quitando marcas de los lugares compartidos, éste no puede diagnosticar correctamente un fallo debido a la falta de información. Este problema se soluciona mediante un protocolo de comunicación entre los diagnosticadores.

El protocolo de comunicación está compuesto por dos algoritmos, ambos estarán en todos los diagnosticadores. El primero actualiza el estado del diagnosticador y genera los mensajes de fallo en el caso que éstos ocurran. El segundo algoritmo está a la espera de algún mensaje de otro diagnosticador. Este algoritmo se puede utilizar para sistemas con M lugares compartidos, con el inconveniente de que el tamaño del mensaje va creciendo con cada fallo, dependiendo del número de lugares en común y de la cantidad de fallos del sistema.

El hecho de que el mensaje se incremente en tamaño es un inconveniente para la comunicación, debido a que el objetivo es la reducción de las transmisiones. Para evitarlo, se propone una longitud fija del mensaje, reduciendo de esta forma la comunicación. Esto se consigue actualizando los protocolos anteriores, de forma que se codifique el mensaje para así obtener una longitud fija.

Caben destacar una serie de ventajas de este método como son el mensaje de tamaño fijo, con el cual se tiene un mayor control sobre la comunicación y que no utiliza modelos de

autómata, haciendo que el modelado del sistema sea más sencillo. Es por esto que tomamos esta técnica como referencia para desarrollar la nuestra.

4.3. Conclusiones / comentarios

Se han descrito tres de las técnicas más usadas, destacando las ventajas e inconvenientes de cada una de ellas. Para nuestro propósito se han considerado interesantes características como la descomposición modular, ya que se asemeja a los sistemas distribuidos, el uso de RdP, puesto que ofrece ventajas en el modelado de sistemas y análisis, y también se ha considerado interesante analizar el flujo compartido, ya que es una característica que afecta a los sistemas que comparten flujo de energía o material.

Capítulo 5

Método de Anidamiento Latente de Fallos (ALf)

El método de Anidamiento Latente de Fallos se utiliza para el análisis de fallos en sistemas complejos, mediante el uso de Redes de Petri Coloreadas (Garcia *et al.*, 2008*b*). El objetivo de esta herramienta es unificar en una sola estructura gráfica los diferentes subsistemas para de esa forma, obtener una mejor interpretación, anidar todos los fallos del sistema en cada lugar de estado y detectar o reconocer el fallo para aislarlo en un lugar específico. Esta metodología se conoce como Redes de Petri Coloreadas para el Diagnóstico de Fallos (RdPCDF), en inglés *Diagnóstic Coloured Petri Nets* (DCPN), usando el método de Anidamiento Latente de Fallos (ALF), en inglés *Fault Latent Nestling Method* (FLNM).

5.1. Definición formal y fundamentos

Las RdPCDF representan los diferentes fallos a diagnosticar mediante colores o marcas.

Definición 4 *Una RdPC para el diagnóstico de fallos se denota:*

$$\mathbb{D} = (P, T, Pre, Post, M_0, C, PLNf, TF, PVf) \tag{5.1}$$

Donde $P, T, Pre, Post, M_0$, tienen las mismas definiciones que para una RdPC, a diferencia de Pre y $Post$ que se dividen cada uno en dos subconjuntos, según sea una transición de comportamiento normal o de fallo.

$$Pre = Pre^T \cup Pre^{TF}$$
$$Post = Post^T \cup Post^{TF}$$

Diferenciando a su vez las funciones en los arcos normales y de fallo, así como la recuperación del fallo.

$Pre = Pre^T : P \times T \to \mathbb{N}$
$Post = Post^T : T \times P \to \mathbb{N}$
$Pre^{TF} : (PLNf \times T_f \cup PVf \times Tr) \to \mathbb{N}$
$Post^{TF} : (T_f \times PVf \cup Tr \times PLNf) \to \mathbb{N}$

C es el conjunto de marcas coloreadas que se divide en los siguientes subconjuntos.

$$C = N \cup f \tag{5.2}$$

Donde N es el conjunto de marcas coloreadas de funcionamiento normal y *f* el conjunto de marcas coloreadas de fallo a diagnosticar.

$PLNf$ son los lugares de anidamiento latente de fallo, PVf son los lugares de verificación de fallo y $TF = T_f \cup T_r$, que es el subconjunto de transiciones de fallo y recuperación.

5.2. Modelado inicial

Primeramente se puede modelar cada subsistema como un modelo de RdP, realizando a continuación un proceso de plegamiento (*folding* en inglés), utilizando las técnicas de RdPC para aprovechar las capacidades que éstas ofrecen y así poder tratar con sistemas complejos de forma mas sencilla.

5.3. Definición del conjunto de fallos

Se debe definir el conjunto de fallos a diagnosticar y asignar a cada uno de ellos unas marcas coloreadas del tipo $f = \{f_1, f_2, ..., f_i\}$.

Cada uno de estos fallos f representa un fallo del sistema, determinado por las condiciones dinámicas del mismo, de forma que, si se verifica el fallo, se garantiza su aislamiento al estar asociado a un tipo de fallo específico de un dispositivo concreto o de una zona del sistema. Pueden haber a su vez fallos de tipo individual f_i o simultáneo $f_i f_k$.

5.4. Lugares de anidamiento latente de fallos (PLN*f*)

Una vez definidos los fallos, se realiza un análisis dinámico del sistema, realizando el anidamiento de fallos en cada uno de los lugares de las RdPC, relacionando las marcas de fallo con

el funcionamiento normal del sistema y depositando en cada lugar P_i una marca coloreada representativa de cada tipo de fallo, de forma que todos los fallos incluidos en el sistema se asignan única y exclusivamente al conjunto de lugares PLN*f*.

Los PLN*f* son los lugares de la RdPCDF en donde se encuentra la marca de fallo f_i, donde permanecerá hasta la activación de una transición de fallo.

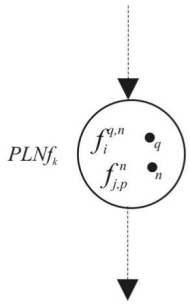


Figura 5.1: Lugar de anidamiento latente de fallos PLNf_k

Se asocia una marca coloreada C_i a cada tipo de fallo f_i, es decir, las marcas se representan en modo texto indicando el tipo de fallo f_i, mientras que las marcas tradicionales (punto negro) representan el comportamiento dinámico normal. En la Figura 5.1 se muestra un ejemplo del anidamiento de fallos y las marcas representativas del comportamiento normal.

5.5. Trayectorias de verificación y recuperación de fallos

Las trayectorias de verificación y recuperación de fallos son las estructuras gráficas que unen un lugar PLN*f* con un lugar PV*f* y viceversa, definidas por las transiciones de fallo (T_f) y recuperación (T_r) respectivamente.

Definición 5 *Una transición de fallo o recuperación en una RdPCDF esta habilitada si cada PLNf_k o PVf en ${}^0TF_j$ cumple la condición:*

para T_f:

$m(PLNf_k) \geq Pre(PLNf_k, T_j)$

para T_r:

$[m(PVf) \geq Pre(PVf, Tr_j)] \wedge [m(PLNf_k) \geq Pre(PLNf_k, Tr_j]$

Cuando se consideran sensores de tipo binario, estos se denominan *SROV*, obteniendo un conjunto de lecturas posibles de:

$$|SROV| = 2^n \tag{5.3}$$

A su vez, el conjunto de $SROV(M_k)$ está dividido en un subconjunto de valores esperados ($SROV_{ev}$) y valores no esperados ($SROV_{nev}$).

$$SROV(M_k) = SROV_{ev}(M_k) \cup SROV_{nev}(M_k) \tag{5.4}$$

Cuando se produce una activación de una lectura no esperada, es decir, se activa una transición de fallo T_f, se expresa como:

$$M(PLNf_k(\langle \bullet q \rangle, \langle f_i^q \rangle))$$
$$[Tf_k/SROV_{nev}(M(PLNf_k(\langle \bullet q \rangle, \langle f_i^q \rangle))) \tag{5.5}$$
$$> M(PVf(\langle f_i^q \rangle))$$

Lo que expresa la Ecuación 5.5 es que un lugar PLNf_k contiene una marca de fallo tipo f_i^q y una marca normal q, cuando se activa la transición de fallo Tf_k, se conduce al marcado $M(PVf(\langle f_i^q \rangle))$, que indica que ha ocurrido el fallo f_i^q.

En algunos casos el fallo se puede recuperar, expresándolo como:

$$M(PVf(\langle f_i^q \rangle))$$
$$[Tr_k/SROV_{ev}(M'(PLNf(\langle \bullet q \rangle)) \wedge M(PVf(\langle f_i^q \rangle))) \tag{5.6}$$
$$> M(PLNf(\langle \bullet q \rangle, \langle f_i^q \rangle))$$

La Ecuación 5.6 expresa que habiendo una marca de fallo f_i^q en el lugar de verificación PVf y una marca normal q en el lugar de anidamiento $PLNf$, cuando se activa la transición de recuperación T_r, se conduce de nuevo al marcado $M(PLNf(\langle \bullet q \rangle, \langle f_i^q \rangle))$.

5.6. Diagnosticabilidad del modelo

Debe existir una trayectoria para cada tipo de fallo desde el lugar de anidamiento latente de fallos $PLNf$ al lugar de verificación de fallos PVf. La capacidad de detección de fallos viene determinada por la capacidad de sensorización. Un sistema puede diagnosticar el fallo q si existe almenos una trayectoria a PVf, expresándolo como:

$$\forall f_i^q \in f \exists (M(PLNf_k(\langle \bullet q \rangle, \langle f_i^q \rangle)))$$
$$[Tf_k/SROV_{nev}(M(PLNf_k(\langle \bullet q \rangle, \langle f_i^q \rangle))) \tag{5.7}$$
$$> M(PVf(\langle f_i^q \rangle))$$

5.7. Ejemplo de la metodología

Para comprender mejor la metodología del Anidamiento Latente de fallos, se presenta un ejemplo básico de una tubería con dos válvulas. El sistema consiste en una tubería para

transportar líquido de un lugar a otro, con dos válvulas a sus extremos y dos sensores a la salida de cada válvula para controlar el flujo. En la Figura 5.2 se presenta la estructura física del sistema.

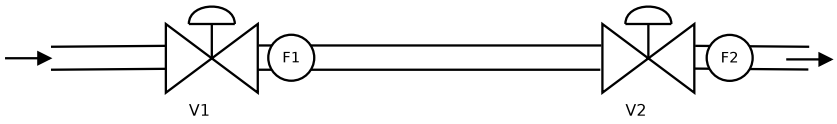


Figura 5.2: Sistema de ejemplo de una tubería con dos válvulas

El funcionamiento del sistema consiste en, para que el líquido fluya de un lugar a otro, deben estar las dos válvulas abiertas, si alguna de las dos válvulas está cerrada, no pasará líquido.

El primer paso es realizar un modelo del sistema de comportamiento normal en una RdP y posteriormente un plegamiento. Debido a la sencillez del ejemplo, se modela directamente una RdPC (Figura 5.3).

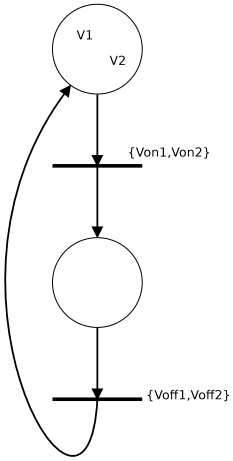


Figura 5.3: RdPC del sistema de una tubería con dos válvulas

A continuación se definen los diferentes fallos que se quieren diagnosticar. En este caso los fallos a diagnosticar son el paso de flujo cuando la válvula esté cerrada (*Fv1a, Fv2a*) y el bloqueo de flujo cuando la válvula se encuentre abierta (*Fv1c, Fv2c*). Estos fallos se detectan con el medidor de flujo que hay a la salida de cada válvula.

Una vez definidos los fallos, se realiza un análisis dinámico para realizar el anidamiento de fallos en cada uno de los lugares de la RdPC, depositando en cada lugar una marca coloreada que represente el tipo de fallo. En la Figura 5.4 se muestra la RdPCDF para el sistema de la tubería con dos válvulas.

El sistema funciona de la siguiente manera, cuando las dos válvulas se encuentren abiertas, el sistema deberá dejar pasar el flujo de líquido, en caso de que el sensor de flujo de una de las dos válvulas no detectara el paso de flujo, se reconocerá un fallo en la válvula en la que no se haya detectado flujo. Cuando las dos válvulas se encuentren cerradas, el sistema

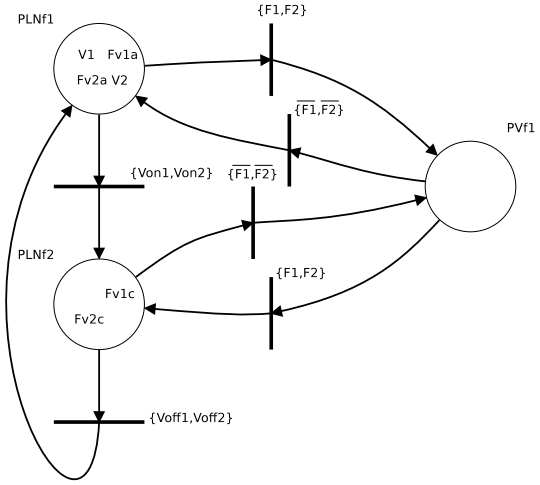


Figura 5.4: RdPCDF del sistema de una tubería con dos válvulas

no deberá dejar pasar flujo de líquido, si el sensor de flujo detecta el paso del mismo, se detectará fallo en la válvula que se haya detectado el paso de flujo.

A continuación se muestra la verificación y recuperación de un fallo, en el que la válvula 1 se encuentra cerrada y el sensor de flujo detecta el paso del mismo. Los demás fallos seguirán la misma estructura.

El aislamiento del fallo *Fv1a* se produce:

$$\begin{gathered} M(PLNf_1(\langle V1\rangle, \langle Fv1a\rangle)) \\ [F1/SROV_{nev}(M(PLNf_1(\langle V1\rangle, \langle Fv1a\rangle))) \\ > M(PVf(\langle Fv1a\rangle)) \end{gathered} \tag{5.8}$$

La recuperación del fallo *Fv1a* se produce:

$$\begin{gathered} M(PVf(\langle Fv1a\rangle)) \\ [\overline{F1}/SROV_{ev}(M(PLNf_1(\langle V1\rangle)) \wedge M(PVf(\langle Fv1a\rangle))) \\ M(PLNf_1(\langle V1\rangle, \langle Fv1a\rangle)) \end{gathered} \tag{5.9}$$

Un ejemplo más formal de esta metodología se presenta en (Rodríguez *et al.*, 2008*a*), aplicándola a un sistema de refrigeración y lubricación de un aerogenerador.

Capítulo 6

Método de Anidamiento Latente de Fallos Distribuido (ALfD)

En la actualidad, muchas aplicaciones son demasiado grandes y complejas para tratarlas como un solo sistema, de ahí la idea de emplear sistemas distribuidos. En este trabajo, partiendo de que el método de Anidamiento Latente de Fallos (ALf) (Garcia *et al.*, 2008*a*) es una herramienta para el análisis de fallos en sistemas complejos centralizados, se pretende ampliarla para su uso en sistemas complejos distribuidos, detectando así los fallos que se producen en los sistemas debido al flujo compartido.

6.1. Estructura e ideas principales

Como se muestra en la Figura 6.1, se parte de un sistema distribuido, el cual se comunica entre sí para realizar las labores de control. Para realizar el diagnóstico se utiliza la técnica de Anidamiento Latente de Fallos, que al tratarse de una técnica centralizada, debe haber un diagnosticador para cada subsistema.

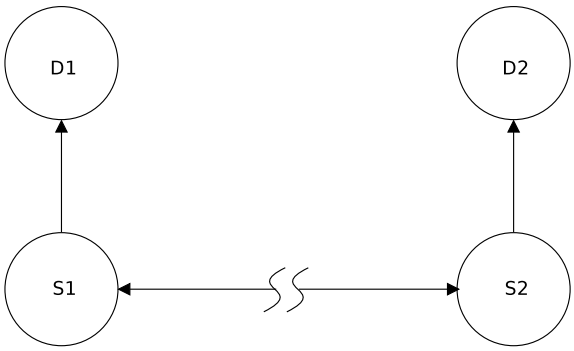


Figura 6.1: Sistema distribuido S1 y S2 con diagnosticadores D1 y D2

En los sistemas distribuidos se pueden producir fallos en los diferentes subsistemas debido al flujo compartido de información y/o material, esto se debe que al producirse un fallo o suceso en un subsistema, éste puede afectar a otro, provocando así un fallo encadenado debido a que no recibe flujo compartido. Para prevenir este tipo de fallos, es necesario que los diagnosticadores se comuniquen entre sí los diferentes fallos y sucesos que se puedan propagar para que de ésta forma, los demás diagnosticadores no realicen un diagnóstico erróneo del fallo producido identificándolo como uno propio.

Un ejemplo sencillo al que se puede aplicar esta técnica es el que se presenta en la Sección 5.7, en el cual el comportamiento de una válvula afecta al comportamiento de la otra, es decir, si una válvula se encuentra cerrada, no habrá flujo de líquido por la tubería, induciendo por tanto un fallo en la otra válvula si se encuentra abierta, ya que no detecta flujo.

Una forma de ampliar la técnica de Anidamiento Latente de Fallos para su uso en sistemas complejos distribuidos es mediante un sistema de comunicación entre diagnosticadores, con esto se consigue evitar que puedan producirse retardos en el sistema de control, consiguiendo así prevenir posibles fallos debido a la comunicación (Figura 6.2).

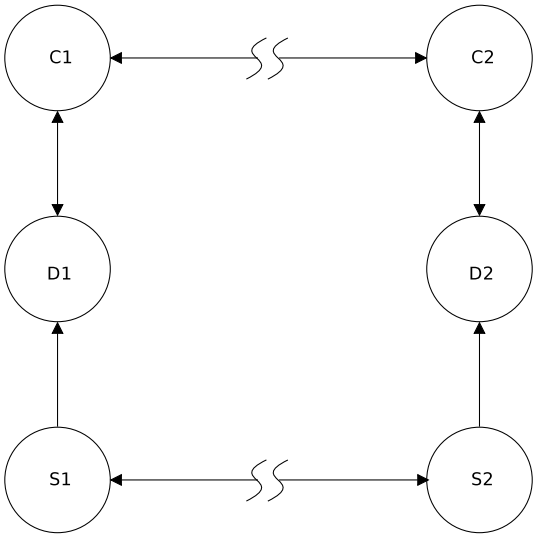


Figura 6.2: Sistema distribuido con diagnosticadores comunicados

Para poder comprender mejor la metodología, se presenta el siguiente ejemplo genérico (Figura 6.3) en el cual se detecta un fallo en el sistema S1, que a su vez, provoca un fallo en el sistema S2.

El sistema está compuesto por dos diagnosticadores y dos lugares de comunicación. En los diagnosticadores se encuentran los lugares de Anidamiento Latente de fallos ($PLNf$) y los lugares de Verificación de fallo (PVf), y en los lugares de Comunicación (Com) es donde se realiza la comunicación entre los dos subsistemas.

La dinámica del sistema es la siguiente, cuando en el diagnosticador 1 se detecte el fallo *fn* a través de la transición *Tfn*, la marca de fallo se moverá del lugar *PLNf1* al lugar *PVf1*, este movimiento da lugar a la activación de la transición *Tcom11*, que moverá la marca *fn*

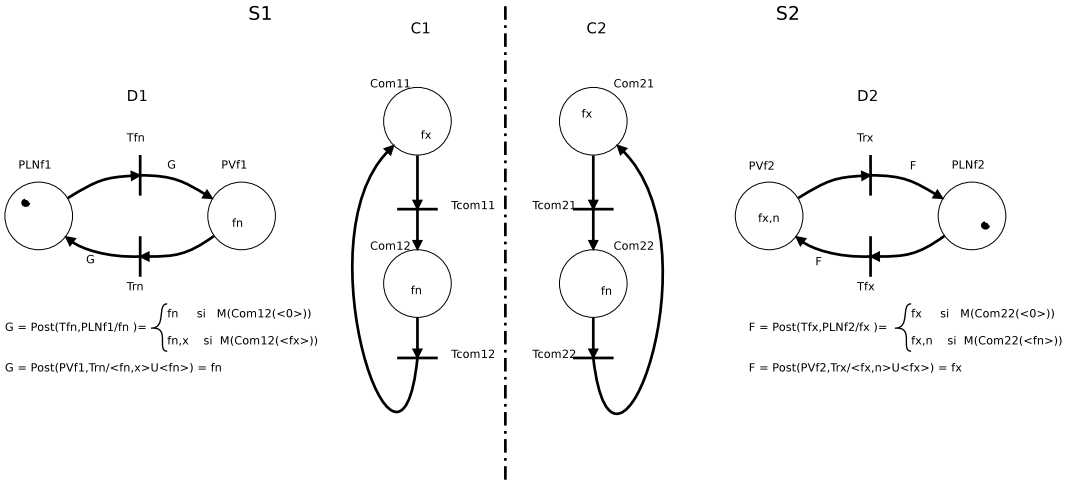


Figura 6.3: Ejemplo genérico de fallo en S1 que provoca un fallo en S2

del lugar de comunicación *Com11* al *Com12*. Este movimiento a su vez activa la transición *Tcom21*, es decir, se realiza la comunicación entre los dos subsistemas, moviendo la marca *fn* del lugar *Com21* al *Com22*. La presencia de la marca *fn* en el lugar *Com22* implica que, cuando el diagnosticador 2 detecte algún fallo a través de la transición *Tfx*, la marca de fallo *fx* se moverá del lugar *PLNf2* al *PVf2*, con la característica que se convertirá en una marca doble, es decir, la marca *fx* se convertirá en *fx,n* al moverse al lugar *PVf2*, indicando que el fallo se ha pruducido como consecuencia del fallo del diagnosticador 1.

La recuperación del fallo se realiza de forma similar, cuando el diagnosticador 1 recupere el fallo a través de la transición *Trn*, la marca de fallo *fn* volverá al lugar *PLNf1*, activando a su vez la transición *Tcom12*, moviendo la marca *fn* del lugar de comunicación *Com12* al lugar *Com11*. Con este movimiento se realiza la comunicación entre los dos subsistemas, activando así la transición *Tcom22*, que moverá la marca del lugar *Com22* al *Com21*. La activación de la transición *Trx*, implica la recuperación del diagnosticador 2, moviendo bien la marca simple *fx* o la marca doble *fx,n* del lugar *PVf2* al lugar de anidamiento *PLNf2*, quedando de esta forma el sistema sin fallos.

Más adelante se realiza una explicación más formal del funcionamiento del sistema.

6.2. Definición formal y fundamentos

Para que los diferentes diagnosticadores se comuniquen entre sí, se emplean los lugares de comunicación. La principal idea de estos lugares es la de comunicar entre los diferentes diagnosticadores solamente los fallos o sucesos que puedan afectar al otro sistema para que de esta forma, se comunique y no se realice un diagnóstico erróneo. Solamente se comunican los fallos o sucesos que afectan o puedan afectar a otro sistema, con el objetivo de mejorar el sistema de comunicación.

Las redes de comunicación se pueden definir como las RdPC, con unas pequeñas modificaciones.

Definición 6 *Una red de comunicación de una RdPCDFD se denota:*

$$\mathbb{C} = (P, T, Pre, Post, M_0, F) \tag{6.1}$$

Donde $P, T, Pre, Post, M_0$, tienen las mismas definiciones que para una RdPC (Sección 5.1), a diferencia de Pre y $Post$ que tienen un argumento adicional que es el color F de disparo de la transición T_j. Así: $Pre(P_i, T_j/F)$, $Post(T_j, P_i/F)$.

F es el conjunto de marcas coloreadas de fallo que se quieren comunicar, pertenecientes al conjunto de marcas coloreadas de fallo a diagnosticar (f), pertenecientes al diagnosticador.

$$F \in f \tag{6.2}$$

Como se puede ver en la Figura 6.4, las redes de comunicación van a tener dos estados, un estado de “fallo” y un estado de “no fallo”.

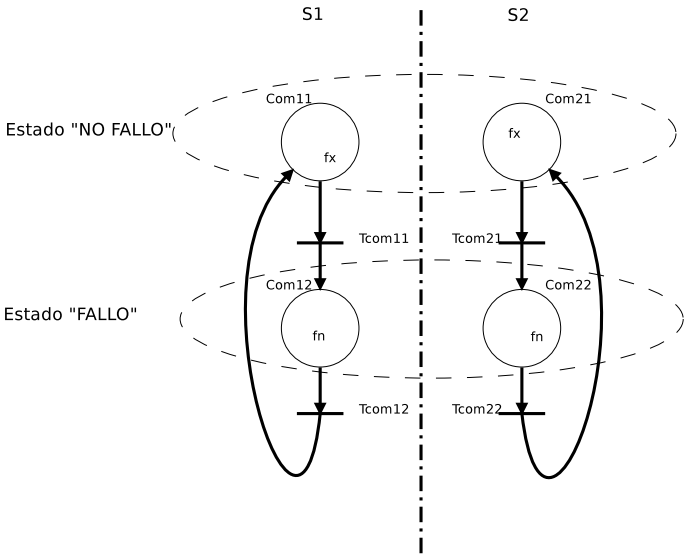


Figura 6.4: Estados de “fallo” y de ”no fallo” en los lugares de comunicación

El estado de “fallo” indica que se ha producido un fallo en un subsistema y éste debe ser comunicado al lugar de comunicación de otro subsistema, para que informe al diagnosticador de que se ha producido un fallo en otro subsistema y lo tenga en cuenta a la hora de realizar el diagnóstico.

El estado de “no fallo” sirve para la recuperación del sistema, comunicando al lugar de comunicación de otro subsistema que se ha recuperado el fallo, y éste a su vez, informe al diagnosticador de la recuperación de dicho fallo.

Estos estados se representan como Com_{jc}, donde j representa el subsistema y c indica el estado en el que se encuentra, 2 para estado de “fallo” y 1 para “no fallo”.

6.3. Definición del conjunto de fallos

Este proceso se realiza de forma muy parecida al método de Anidamiento Latente de fallos. El primer paso es definir el conjunto de fallos $F = \{F_1, F_2, ..., F_i\}$ que se van a comunicar entre los diagnosticadores, para asignarlos a los diferentes lugares de comunicación y de esta forma poder transmitirlos a los subsistemas.

Cada uno de estos fallos F representa un fallo o suceso de un subsistema que se va a transmitir a los demás diagnosticadores. Este fallo solamente se definirá en los lugares de comunicación a los que vaya a afectar, esto quiere decir que si se produce un fallo fp en el subsistema S_a y el fallo no afecta al subsistema S_b, no será necesario definir el fallo fp en este subsistema.

Para el ejemplo genérico que se propone (Figura 6.3) se definen los fallos fn y fx, uno para cada diagnosticador.

En la Sección 5.7, como ya se ha mencionado, se presenta un ejemplo al que se puede aplicar este nuevo método, los fallos a transmitir son los fallos inducidos al encontrarse una válvula cerrada y otra abierta ($Fi1c, Fi2c$), que provocan un fallo inducido por la válvula cerrada en la válvula que se encuentre abierta al no haber flujo.

6.4. Modificaciones en los diagnosticadores

En los diferentes subsistemas se utiliza la técnica de Anidamiento Latente de fallos, a la cual, es necesario realizar unas pequeñas modificaciones en los diagnosticadores. Las modificaciones consisten en añadir una función $Post$ a las transiciones de fallo y recuperación, de forma que se puedan modificar las marcas según las condiciones de disparo de la transición.

Definición 7 *Las funciones de los arcos se definen:*

$$Post(Tf_p, PLNf_j/fp) = \begin{cases} fp & si \quad M(Com_{j2}(<0>)) \\ fp, q & si \quad M(Com_{j2}(<fq>)) \end{cases}$$

$$Post(PVf_j, Tr_p/ <fp, q> \cup <fp>) = fp$$

La función en la transición de fallo indica que cuando haya una marca de fallo fq en el lugar de comunicación que recibe el fallo (Com_{j2}), una marca de fallo fp en el lugar $PLNf_j$ y se active la transición de fallo Tf_p, la función convertirá la marca de fallo simple fp en una marca doble fp, q, indicando que el fallo de fp se ha producido a consecuencia del fallo fq.

La función en la transición de recuperación indica que cuando esté la marca fp, q o fp en PVf_j y se active la transición Tr_p, la marca se convertirá en fp de nuevo en el lugar $PLNf_j$.

En el ejemplo genérico propuesto (Figura 6.3), una vez definidos los fallos que se van a comunicar y los lugares de comunicación, se deben modificar los diagnosticadores para que éstos modifiquen las marcas de fallo según el disparo de las transiciones. En este ejemplo solamente se definen las funciones del subsistema 2, ya que para los demás subsistemas se realizaría de la misma forma.

$$Post(Tf_x, PLNf_2/fx) = \begin{cases} fx & si \quad M(Com_{22}(< 0 >)) \\ fx, n & si \quad M(Com_{22}(< fn >)) \end{cases}$$

$$Post(PVf_2, Tr_x/ < fx, n > \cup < fx >) = fx$$

Como se puede ver en la Figura 6.5, las funciones anteriores indican que si hay una marca fx en el lugar $PLNf_2$, una marca fn en Com_{22} y se activa la transición Tf_x, la función convertirá la marca fx en una marca fx, n en el lugar PVf_2. Si hay una marca fx, n o una marca fx en el lugar PVf_2 y se activa la transición Tr_x, se pondrá la marca fx en el lugar $PLNf_2$.

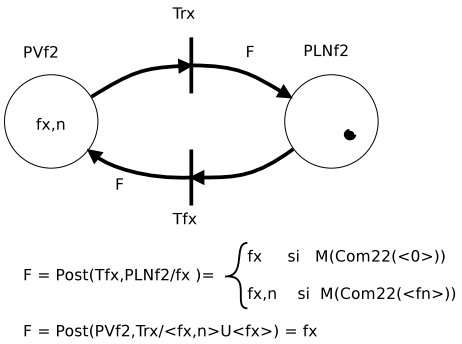


Figura 6.5: Funciones de las transiciones de fallo del subsistema 2

Una modificación que va a sufrir el ejemplo de la Sección 5.7 para poder aplicar este nuevo método es la de modificar los diagnosticadores, para que modifiquen las marcas de fallo según las condiciones de disparo de la transición.

Cuando una válvula se encuentre cerrada y otra abierta, en la válvula abierta habrá un fallo inducido por la válvula cerrada que provoca que no haya flujo de líquido. La modificación se realiza para los dos diagnosticadores de la misma forma.

Modificación para el diagnosticador 2:

$$Post(TF4, PLNf_4/Fv2c) = \begin{cases} Fv2c & si \quad M(Com_{22}(< 0 >)) \\ Fv2c, Fi1c & si \quad M(Com_{22}(< Fi1c >)) \end{cases}$$

$$Post(PVf_2, TR4/ < Fv2c, Fi1c > \cup < Fv2c >) = Fv2c$$

Modificación para el diagnosticador 1:

$$Post(TF2, PLNf_2/Fv1c) = \begin{cases} Fv1c & si \quad M(Com_{12}(<0>)) \\ Fv1c, Fi2c & si \quad M(Com_{12}(<Fi2c>)) \end{cases}$$

$$Post(PVf_1, TR2/ < Fv1c, Fi2c > \cup < Fv1c >) = Fv1c$$

6.5. Trayectorias de comunicación de fallos

Las trayectorias de comunicación de fallos son las encargadas de realizar la comunicación entre los diferentes diagnosticadores y los lugares de comunicación, y viceversa, comunicándose también entre sí los lugares de comunicación. Las transiciones que unen los diferentes lugares se definen mediante la aparición de marcas en los lugares de los diagnosticadores y de los lugares de comunicación.

Las trayectorias de comunicación de fallos de los lugares de comunicación del sistema que detecta el fallo se definen:

Definición 8 *Trayectoria de fallo del lugar de comunicación que detecta el fallo*

$$M(Com_{j1}(<f_p>))$$

$$[Tcom_{j1}/SROV_{nev}(M(Com_{j1}(<f_p>)))$$

$$> M(Com_{j2}(<f_p>))$$

Donde $Tcom_{j1}$ es la transición que se activa al aparecer la marca fp en el lugar PVf_j del diagnosticador j.

Definición 9 *Trayectoria de recuperación del lugar de comunicación que detecta el fallo*

$$M(Com_{j2}(<f_p>))$$

$$[Tcom_{j2}/SROV_{ev}(M(Com_{j2}(<f_p>)))$$

$$> M(Com_{j1}(<f_p>))$$

Donde $Tcom_{j2}$ es la transición que se activa al desaparecer la marca fp del lugar PVf_j del diagnosticador j.

A continuación se van a definir las trayectorias de comunicación de fallos de los lugares de comunicación del sistema al que se le comunica el fallo. Este punto es en el que se realiza la comunicación entre los dos subsistemas.

Definición 10 *Trayectoria de fallo del lugar de comunicación que recibe el fallo*

$$M(Com_{j1}(< f_p >))$$

$$[Tcom_{j1}/SROV_{nev}(M(Com_{j1}(< f_p >)))$$

$$> M(Com_{j2}(< f_p >))$$

Donde $Tcom_{j1}$ es la transición que se activa al aparecer la marca fp en Com_{j2} del diagnosticador que comunica el fallo.

Definición 11 *Trayectoria de recuperación del lugar de comunicación que recibe el fallo*

$$M(Com_{j2}(< f_p >))$$

$$[Tcom_{j2}/SROV_{ev}(M(Com_{j2}(< f_p >)))$$

$$> M(Com_{j1}(< f_p >))$$

Donde $Tcom_{j2}$ es la transición que se activa al aparecer la marca fp en Com_{j1} del el diagnosticador que recibe el fallo.

Para el ejemplo genérico propuesto en la Figura 6.3 las trayectorias de verificación, recuperación y comunicación de fallos serían las siguientes.

Las trayectorias de los lugares de comunicación del sistema 1, el que detecta el fallo, se definen:

- Trayectoria de fallo del lugar de comunicación del sistema 1

$$M(Com_{11}(< f_n >))$$

$$[Tcom_{11}/SROV_{nev}(M(Com_{11}(< f_n >)))$$

$$> M(Com_{12}(< f_n >))$$

  Donde $Tcom_{11}$ es la transición que se activa al aparecer la marca fn en el lugar PVf_1 del diagnosticador 1.

- Trayectoria de recuperación del lugar de comunicación del sistema 1

$$M(Com_{12}(< f_n >))$$

$$[Tcom_{12}/SROV_{ev}(M(Com_{12}(< f_n >)))$$

$$> M(Com_{11}(< f_n >))$$

  Donde $Tcom_{12}$ es la transición que se activa al desaparecer la marca fn del lugar PVf_1 del diagnosticador 1.

Seguidamente se definen las diferentes trayectorias de comunicación de fallos de los lugares de comunicación del sistema 2, al que se le comunica el fallo.

– Trayectoria de fallo del lugar de comunicación del sistema 2

$$M(Com_{21}(< f_n >))$$

$$[Tcom_{21}/SROV_{nev}(M(Com_{21}(< f_n >)))$$

$$> M(Com_{22}(< f_n >))$$

Donde $Tcom_{21}$ es la transición que se activa al aparecer la marca fn en Com_{12}.

– Trayectoria de recuperación del lugar de comunicación del sistema 2

$$M(Com22(< f_n >))$$

$$[Tcom_{22}/SROV_{ev}(M(Com_{22}(< f_n >)))$$

$$> M(Com_{21}(< f_n >))$$

Donde $Tcom_{22}$ es la transición que se activa al aparecer la marca fn en Com_{11}.

Las trayectorias de verificación, recuperación y comunicación de fallos del ejemplo de la Sección 5.7 serán las siguientes, mostrando el esquema completo en la Figura 6.6:

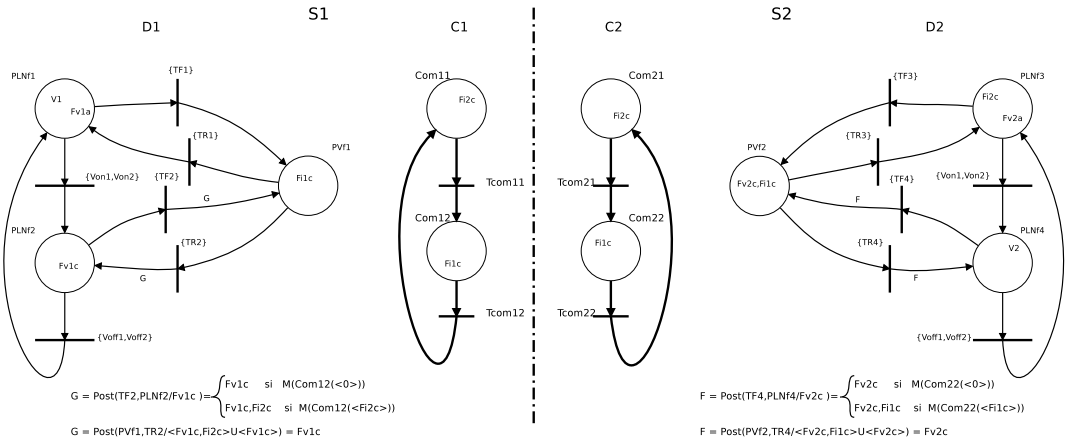


Figura 6.6: Ejemplo completo del método distribuido para el ejemplo de la Sección 5.7

Las trayectorias se definen de la misma forma para comunicar un fallo de un subsistema a otro, a continuación se definen solamente las trayectorias de los lugares de comunicación del subsistema 1, cuando éste detecta el fallo, ya que cuando se detecta el fallo en el subsistema 2, la comunicación se realiza de la misma forma:

Cuando aparezca la marca *Fi1c* en el lugar PVf_1 del diagnosticador 1, se activará la transición $Tcom_{11}$, moviendo la marca *Fi1c* del lugar Com_{11} al lugar Com_{12}.

Una vez la marca *Fi1c* se encuentre en el lugar Com_{12}, se realiza la comunicación entre los diferentes sistemas, activando la transición $Tcom_{21}$, que mueve la marca *Fi1c* del lugar $Tcom_{21}$ al lugar $Tcom_{22}$, donde el diagnosticador 2 ya recibe el fallo.

La recuperación del fallo se realiza cuando desaparece la marca *Fi1c* del lugar PVf_1 del diagnosticador 1, que mueve la marca *Fi1c* del lugar Com_{12} al lugar Com_{11} auna vez se active la transición $Tcom_{12}$.

Al recuperarse el diagnosticador 1 y encontrarse la marca *Fi1c* en el lugar Com_{11}, se realiza la comunicación para activar la $Tcom_{22}$ y mover la marca *Fi1c* del lugar Com_{22} al lugar Com_{21}.

6.6. Conclusiones del método

Este método como ya se ha comentado, ofrece la ventaja de diagnosticar los fallos que se producen debido al flujo compartido en sistemas complejos distribuidos. Partiendo de la base del método de Anidamiento Latente de Fallos (ALf), se ha conseguido evitar que se realice un diagnóstico erróneo del fallo producido debido al flujo compartido mediante la comunicación entre los diagnosticadores.

Sin embargo, dicho método presenta el inconveniente de que se pierde poder de diagnóstico, ya que al realizar la comunicación, el diagnosticador que recibe el fallo depende del anterior, no pudiendo de esta forma diagnosticar un fallo propio mientras el otro se encuentre activo.

Una mejora para esta técnica podría ser el aplicar un retardo en la propagación y recuperación del fallo, ya que dependiendo del sistema, el fallo puede tardar un tiempo en propagarse y afectar a los demás sistemas. Ésta mejora no evita el problema, ya que es una característica de los sistemas, pero disminuye el tiempo en que se está detectando el fallo en el diagnosticador que recibe el fallo. En el Capítulo 7 se propone una solución para esta mejora.

Capítulo 7

Método de Anidamiento Latente de Fallos Distribuido (ALfD) con retardo

El Método de Anidamiento Latente de Fallos Distribuido (ALfD) presentado en el Capítulo 6 es una herramienta para el análisis de fallos en sistemas complejos distribuidos, detectando los fallos que se producen a causa del flujo compartido. En la mayoría de estos casos, los fallos de flujo compartido tienen un tiempo de propagación antes de que puedan afectar al siguiente sistema, de ahí la necesidad de tener en cuenta este tiempo.

7.1. Estructura e ideas principales

En los sistemas complejos distribuidos, los fallos suelen tener un tiempo de propagación antes de afectar al siguiente sistema, esto se debe al flujo compartido existente entre los diferentes sistemas. Este tiempo de propagación se debe tener en cuenta para saber con mayor certeza el momento en el que el fallo va a afectar al siguiente sistema y por tanto tener más información para no hacer un diagnóstico erróneo del fallo producido. El tiempo de propagación y recuperación se ha de asignar de forma manual en la puesta en marcha del proceso o con la ayuda de un experto.

El ejemplo que se presenta en la Sección 5.7 cumple con esta característica, en el que el comportamiento de una válvula afecta al comportamiento de la otra, pasando un tiempo entre la detección del fallo y la inducción del mismo en la otra válvula. Esto es debido a que el flujo de líquido tarda un tiempo en propagarse de una válvula a otra.

La información del tiempo de propagación de fallo se encuentra en los lugares de comunicación, los cuales comunicarán los fallos al siguiente subsistema cuando este tiempo se cumpla, de forma que los demás diagnosticadores puedan realizar el diagnóstico con mayor exactitud.

El sistema está compuesto de la misma forma que para el método de Anidamiento Latente de Fallos Distribuido (ALfD), por un diagnosticador y un sistema de comunicación para cada subsistema.

En la Figura 7.1 se puede ver la ampliación del ejemplo genérico anterior (Capítulo 6) en el que ahora se tiene en cuenta el tiempo de propagación de los fallos.

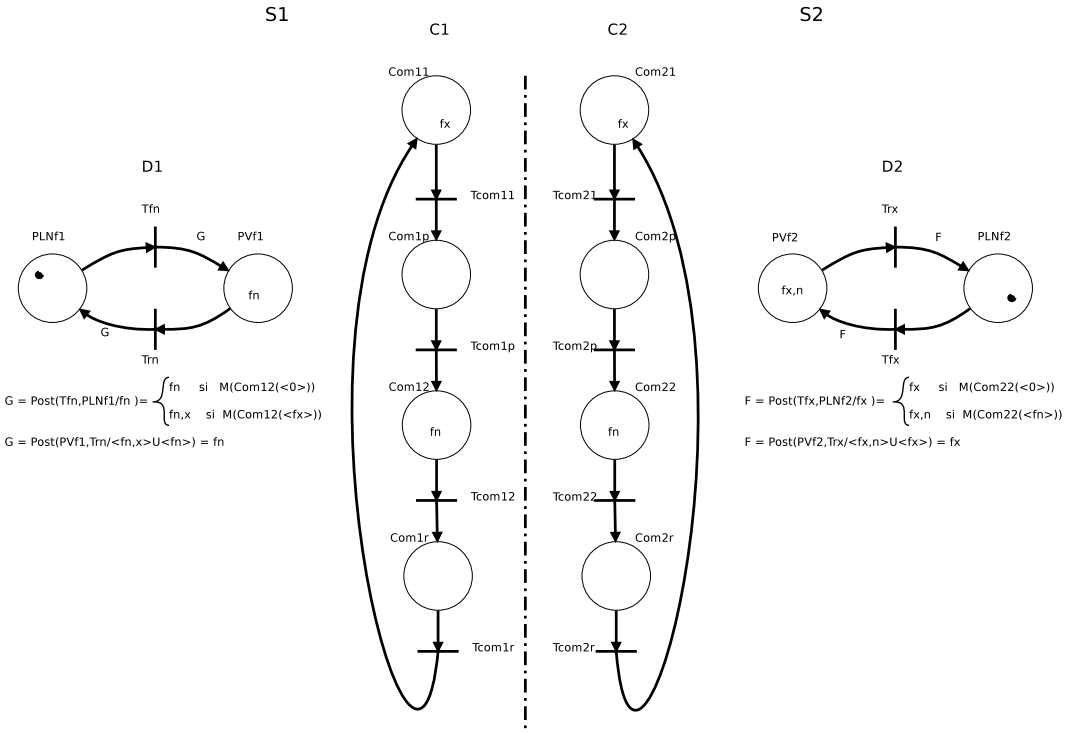


Figura 7.1: Ejemplo de fallo en S1 que provoca fallo en S2 con retardo en la propagación

La dinámica del sistema es muy parecida a la del ejemplo del Capítulo 6, añadiendo en este caso los lugares de propagación y recuperación para así tener en cuenta el tiempo de propagación y recuperación de fallo.

Cuando el diagnosticador 1 detecte el fallo *fn* mediante la transición *Tfn*, la marca se moverá de *PLNf1* a *PVf1*, activando así la transición *Tcom11*, que moverá la marca *fn* del lugar *Com11* al *Com1p*, activando de esta forma el temporizador de la transición *Tcom1p*. Una vez transcurrido el tiempo de propagación del fallo, la marca *fn* pasará del lugar *Com1p* al lugar *Com12*, activando a su vez la transición *Tcom21*, realizando así la comunicación entre los dos subsistemas, y moviendo la marca *fn* de *Com21* a *Com2p*. En este caso al haberse cumplido el tiempo de propagación la transición *Tcom2p* estará activada, moviendo la marca *fn* del lugar *Com2p* a *Com22*. La presencia de esta marca en el lugar *Com22* implica que, cuando el diagnosticador 2 detecte algún fallo mediante la transición *Tfx*, la marca *fx* se moverá del lugar *PLNf2* al lugar *PVf2*, convirtiéndose en una marca doble *fx,n*, indicando que se ha producido el fallo en el diagnosticador 2 como consecuencia del fallo del diagnosticador 1.

La recuperación del fallo se realiza cuando el diagnosticador 1 recupere al fallo a través de la transición *Trn*, devolviendo la marca *fn* al lugar *PLNf1*, activando así la transición *Tcom12* que moverá la marca *fn* del lugar de comunicación *Com12* al lugar *Com1r*. Con la marca *fn* en el lugar *Com1r* se activará el temporizador de la transición *Tcom1r*. Al cumplirse el tiempo de recuperación, la marca *fn* se moverá al lugar *Com11*, realizando de esta forma la comunicación entre los dos subsistemas para activar la transición *Tcom22*, que moverá la marca del lugar *Com22* al lugar *Com2r*. Al haberse cumplido el tiempo de recuperación, la transición *Tcom2r* estará activa, moviendo la marca *fn* al lugar *Com21*, quedando así el sistema sin ningún fallo detectado provocado por el diagnosticador 1.

Cuando se recupere el fallo del diagnosticador 2 mediante la transición *Trx*, la marca simple *fx* o la marca doble *fx,n* se moverán al lugar de anidamiento *PLNf2* como *fx*.

Se realiza una explicación más formal más adelante.

7.2. Definición formal y fundamentos

Al igual que en una RdPCDFD, solamente se transmiten los fallos que afectan a otro sistema, con la diferencia de que estos llevan asociado un tiempo de propagación y recuperación.

Este método se define igual que las RdPCDFD con el añadido del tiempo de propagación.

Definición 12 *Una red de comunicación de una RdPCDFDR se denota:*

$$\mathbb{CR} = (P, T, Pre, Post, M_0, F, t) \tag{7.1}$$

Donde $P, T, Pre, Post, M_0$ y F, tienen las mismas definiciones que para una RdPCDFD.

t es el tiempo de propagación y recuperación de fallo que se le asigna a cada tipo de fallo.

$$t = tp \cup tr \tag{7.2}$$

Como se puede ver en la Figura 7.2, los lugares de comunicación van a tener cuatro estados, estado de “fallo”, estado de “no fallo”, estado de “propagación” y estado de “recuperación”.

El estado de “fallo” y “no fallo” se comportan de la misma forma que en el método de Anidamiento Latente de Fallos Distribuido (ALfD), comunicar la ocurrencia y recuperación del fallo a los demás subsistemas.

El estado de “propagación” realiza una espera hasta que se cumpla el tiempo de propagación del fallo, una vez se cumpla este tiempo, significa que el fallo va a afectar al siguiente subsistema y por tanto se cambia al estado de “fallo”.

En el estado de “recuperación”, al igual que el estado de “propagación”, realiza la espera hasta que se cumpla el tiempo de recuperación, pasado ese tiempo, significa que el fallo ya no va a afectar al siguiente subsistema y se cambia al estado de “no fallo”.

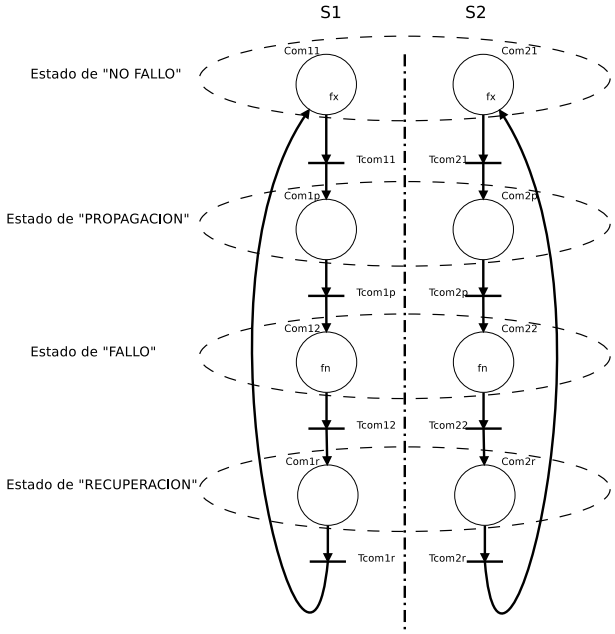


Figura 7.2: Estados de “fallo”, “no fallo”, “propagación” y “recuperación”

Los estados se representan como Com_{jc}, donde j representa el subsistema al que pertenece el lugar de comunicación y c indica el estado en el que se encuentra, representando con *1* para “no fallo”, *2* para “fallo”, *p* para “propagación” y *r* para “recuperación”.

7.3. Definición del conjunto de fallos y tiempo de propagación/recuperación

La definición de los fallos se realiza de la misma forma que para el método de Anidamiento Latente de fallos distribuido, primero se define el conjunto de fallos que se van a comunicar $F = \{F_1, F_2, ..., F_i\}$, y una vez definidos, se ha de definir el tiempo de propagación y recuperación t para cada uno de ellos.

Cada uno de los fallos F tiene asociado un tiempo t, el cual indica el tiempo entre que es detectado el fallo hasta que éste afecta a otro subsistema. Cada tiempo de propagación es diferente, ya que cada fallo puede tardar más o menos tiempo en propagarse.

El tiempo de recuperación será ligeramente superior al tiempo de propagación, debido a que el diagnosticador ha de estar detectando el fallo un tiempo antes de que se propague y una vez propagado este.

Para el ejemplo propuesto se definen los fallos *fn* y *fx*, uno para cada diagnosticador. Se ha considerado un tiempo de propagación de 5 segundos y un tiempo de recuperación de 7 segundos. Este tiempo se ha elegido al azar, ya que se trata de un ejemplo.

7.4. Modificaciones en los diagnosticadores

No es necesario realizar ninguna modificación respecto al método de Anidamiento Latente de fallos distribuido, debido a que no se han modificado los lugares de comunicación de forma que afecten a los diagnosticadores. La modificación es la misma, la función $Post$ de las transiciones de fallo y recuperación de los diagnosticadores, así que solamente se muestra la función genérica de los arcos de fallo y recuperación.

Definición 13 *Las funciones de los arcos se definen:*

$$Post(Tf_a, PLNf_j/fa) = \begin{cases} fa & si \quad M(Com_{j2}(< 0 >)) \\ fa, b & si \quad M(Com_{j2}(< fb >)) \end{cases}$$

$$Post(PVf_j, Tr_a/ < fa, b > \cup < fa >) = fa$$

La función en la transición de fallo indica que cuando haya una marca de fallo fb en el lugar de comunicación que recibe el fallo (Com_{j2}), una marca de fallo fa en el lugar $PLNf_j$ y se active la transición de fallo Tf_a, la función convertirá la marca de fallo simple fa en una marca doble fa, b, indicando que el fallo de fa se ha producido como consecuencia del fallo fb.

La función en la transición de recuperación indica que cuando esté la marca fa, b o la marca fa en PVf_j y se active la transición Tr_a, la marca se convertirá en fa de nuevo en el lugar $PLNf_j$.

7.5. Trayectorias de comunicación de fallos

Al igual que en el Capítulo 6, las trayectorias de comunicación de fallos realizan la comunicación entre los diferentes diagnosticadores y los lugares de comunicación, y viceversa. Las transiciones se activan mediante la aparición de marcas en diferentes lugares de los lugares de comunicación y de los diagnosticadores.

Las trayectorias de los lugares de comunicación del sistema que detecta el fallo se definen:

Definición 14 *Trayectoria de fallo del lugar de comunicación que detecta el fallo*

$$M(Com_{j1}(< f_a >))$$

$$[Tcom_{j1}/SROV_{nev}(M(Com_{j1}(< f_a >)))$$

$$> M(Com_{jp}(< f_a >))$$

Donde $Tcom_{j1}$ es la transición que se activa al aparecer la marca fa en el lugar PVf_j del diagnosticador j.

Definición 15 *Trayectoria de fallo del lugar de comunicación que propaga el fallo*

$$M(Com_{jp}(< f_a >))$$

$$[Tcom_{jp}/SROV_{nev}(M(Com_{jp}(< f_a >)))$$

$$> M(Com_{j2}(< f_a >))$$

Donde $Tcom_{jp}$ es la transición temporizada que se activa al aparecer la marca fa en el lugar Com_{jp}, donde el tiempo se define como tiempo de propagación.

Definición 16 *Trayectoria de recuperación del lugar de comunicación que detecta el fallo*

$$M(Com_{j2}(< f_a >))$$

$$[Tcom_{j2}/SROV_{ev}(M(Com_{j2}(< f_a >)))$$

$$> M(Com_{jr}(< f_a >))$$

Donde $Tcom_{j2}$ es la transición que se activa al desaparecer la marca fa del lugar PVf_j del diagnosticador j.

Definición 17 *Trayectoria de recuperación del lugar de comunicación que recupera el fallo*

$$M(Com_{jr}(< f_a >))$$

$$[Tcom_{jr}/SROV_{nev}(M(Com_{jr}(< f_a >)))$$

$$> M(Com_{j1}(< f_a >))$$

Donde $Tcom_{jr}$ es la transición temporizada que se activa al aparecer la marca fa en el lugar Com_{jr}, donde el tiempo se define como tiempo de recuperación.

Una vez definidas las trayectorias de comunicación de fallos del sistema que detecta el fallo, se van a definir las trayectorias del sistema al que se le comunica el fallo. Es en esta punto en el que se realiza la comunicación entre los diferentes subsistemas.

Definición 18 *Trayectoria de fallo del lugar de comunicación que recibe el fallo*

$$M(Com_{j1}(< f_a >))$$

$$[Tcom_{j1}/SROV_{nev}(M(Com_{j1}(< f_a >)))$$

$$> M(Com_{jp}(< f_a >))$$

Donde $Tcom_{j1}$ es la transición del sistema que recibe el fallo que se activa al aparecer la marca fa en Com_{j2} del sistema que detecta el fallo.

Definición 19 *Trayectoria de fallo del lugar de comunicación que propaga el fallo*

$$M(Com_{jp}(< f_a >))$$

$$[Tcom_{jp}/SROV_{nev}(M(Com_{jp}(< f_a >)))$$

$$> M(Com_{j2}(< f_a >))$$

Donde $Tcom_{jp}$ es la transición temporizada que se activa al aparecer la marca fa en el lugar Com_{jp} del sistema que recibe el fallo, donde el tiempo se define como tiempo de propagación. En este caso el tiempo ya se ha cumplido por tanto la transición se encontrará activa.

Definición 20 *Trayectoria de recuperación del lugar de comunicación que recibe el fallo*

$$M(Com_{j2}(< f_a >))$$

$$[Tcom_{j2}/SROV_{ev}(M(Com_{j2}(< f_a >)))$$

$$> M(Com_{jr}(< f_a >))$$

Donde $Tcom_{j2}$ es la transición del sistema que recibe el fallo que se activa al aparecer la marca fa en Com_{j1} del sistema que detecta el fallo.

Definición 21 *Trayectoria de recuperación del lugar de comunicación que recupera el fallo*

$$M(Com_{jr}(< f_a >))$$

$$[Tcom_{jr}/SROV_{nev}(M(Com_{jr}(< f_a >)))$$

$$> M(Com_{j1}(< f_a >))$$

Donde $Tcom_{jr}$ es la transición temporizada que se activa al aparecer la marca fa en el lugar Com_{jr} del sistema que recibe el fallo, donde el tiempo se define como tiempo de recuperación. El tiempo de recuperación ya se ha cumplido en este caso, de modo que la transición se encontrá activa.

Las trayectorias de verificación, recuperación y comunicación de fallos para el ejemplo de la Figura 7.1 serían las siguientes.

Las trayectorias de los lugares de comunicación que detecta el fallo, subsistema 1, se definen:

- Trayectoria de fallo del lugar de comunicación del sistema 1

  $$M(Com_{11}(< f_n >))$$

  $$[Tcom_{11}/SROV_{nev}(M(Com_{11}(< f_n >)))$$

  $$> M(Com_{1p}(< f_n >))$$

  Donde $Tcom_{11}$ es la transición que se activa al aparecer la marca fn en el lugar PVf_1 del diagnosticador 1.

– Trayectoria de fallo del lugar de comunicación del sistema 1 que propaga el fallo

$$M(Com_{1p}(< f_n >))$$

$$[Tcom_{1p}/SROV_{nev}(M(Com_{1p}(< f_n >)))$$

$$> M(Com_{12}(< f_n >))$$

Donde $Tcom_{1p}$ es la transición temporizada que se activa al aparecer la marca fn en el lugar Com_{1p}, donde el tiempo se define como tiempo de propagación.

– Trayectoria de recuperación del lugar de comunicación del sistema 1

$$M(Com_{12}(< f_n >))$$

$$[Tcom_{12}/SROV_{ev}(M(Com_{12}(< f_n >)))$$

$$> M(Com_{1r}(< f_n >))$$

Donde $Tcom_{12}$ es la transición que se activa al desaparecer la marca fn del lugar PVf_1 del diagnosticador 1.

– Trayectoria de recuperación del lugar de comunicación del sistema 1 que recupera e fallo

$$M(Com_{1r}(< f_n >))$$

$$[Tcom_{1r}/SROV_{nev}(M(Com_{1r}(< f_n >)))$$

$$> M(Com_{11}(< f_n >))$$

Donde $Tcom_{1r}$ es la transición temporizada que se activa al aparecer la marca fn en el lugar Com_{1r}, donde el tiempo se define como tiempo de recuperación.

A continuación se definen las diferentes trayectorias de comunicación de fallos de los lugares de comunicación al sistema que se le comunica el fallo, el sistema 2.

– Trayectoria de fallo del lugar de comunicación del sistema 2

$$M(Com_{21}(< f_n >))$$

$$[Tcom_{21}/SROV_{nev}(M(Com_{21}(< f_n >)))$$

$$> M(Com_{2p}(< f_n >))$$

Donde $Tcom_{21}$ es la transición que se activa al aparecer la marca fn en Com_{12}.

– Trayectoria de fallo del lugar de comunicación del sistema 2 que propaga el fallo

$$M(Com_{2p}(< f_n >))$$

$$[Tcom_{2p}/SROV_{nev}(M(Com_{2p}(< f_n >)))$$

$$> M(Com_{22}(< f_n >))$$

Donde $Tcom_{2p}$ es la transición temporizada que se activa al aparecer la marca fn en el lugar Com_{2p}, donde el tiempo se define como tiempo de propagación. El tiempo de propagación ya se ha cumplido en este caso, de modo que la transición se encontrá activa.

– Trayectoria de recuperación del lugar de comunicación del sistema 2

$$M(Com_{22}(< f_n >))$$

$$[Tcom_{22}/SROV_{ev}(M(Com_{22}(< f_n >)))$$

$$> M(Com_{2r}(< f_n >))$$

Donde $Tcom_{22}$ es la transición que se activa al aparecer la marca fn en Com_{11}.

– Trayectoria de recuperación del lugar de comunicación del sistema 2 que recupera el fallo

$$M(Com_{2r}(< f_n >))$$

$$[Tcom_{2r}/SROV_{nev}(M(Com_{2r}(< f_n >)))$$

$$> M(Com_{21}(< f_n >))$$

Donde $Tcom_{2r}$ es la transición temporizada que se activa al aparecer la marca fn en el lugar Com_{2r}, donde el tiempo se define como tiempo de recuperación. En este caso el tiempo ya se ha cumplido por tanto la transición se encontrará activa.

Para el ejemplo de la Sección 5.7 las trayectorias de verificación, recuperación y comunicación serán las siguientes. El esquema completo se muestra en la Figura 7.3.

A continuación se definen las trayectorias de los lugares de comunicación cuando el subsistema 1 detecta el fallo, para el subsistema 2, se realiza de la misma forma cuando se detecta el fallo.

Cuando aparezca la marca *Fi1c* en el lugar PVf_1 del diagnosticador 1, se activará la transición $Tcom_{11}$, que moverá la marca *Fi1c* del lugar Com_{11} al lugar Com_{1p}.

Una vez esté la marca *Fi1c* en el lugar Com_{1p}, se activará la transición $Tcom_{1p}$, siendo una transición temporizada definida con el tiempo estimado de propagación. Una vez transcurrido este tiempo, moverá la marca *Fi1c* al lugar Com_{12}.

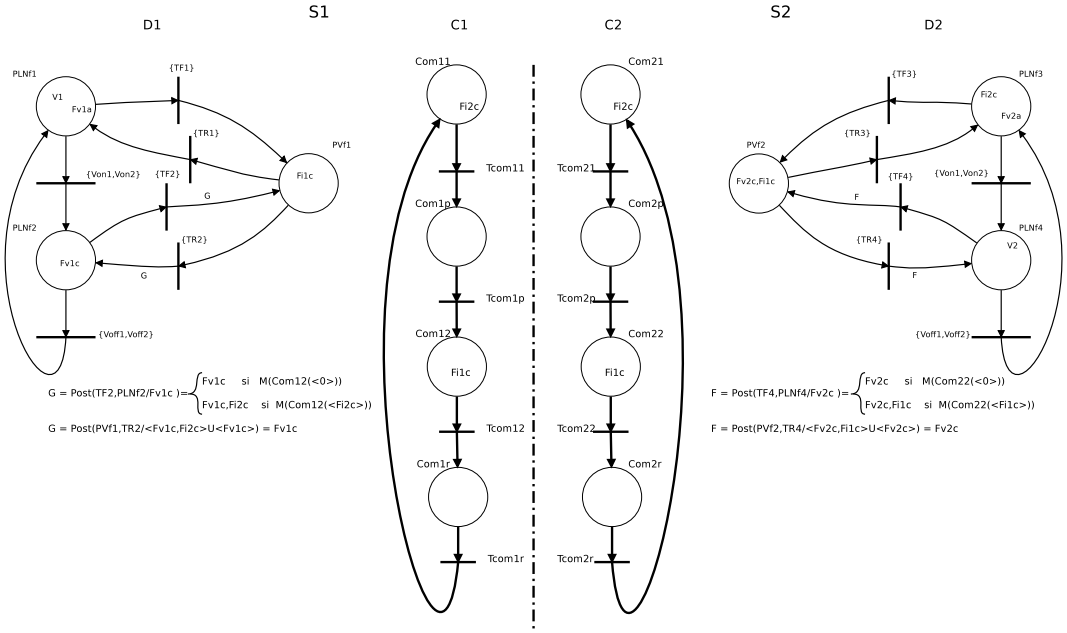


Figura 7.3: Ejemplo método distribuido con retardo para el ejemplo de la Sección 5.7

Cuando la marca *Fi1c* se encuentre en el lugar Com_{12}, se realiza la comunicación al siguiente sistema, activando la transición $Tcom_{21}$, que moverá la marca *Fi1c* del lugar Com_{21} al lugar Com_{2p}.

Al aparecer la marca *Fi1c* en el lugar Com_{2p}, se activa la transición $Tcom_{2p}$, siendo esta transición temporizada definida por el tiempo estimado de propagación. Este tiempo ya se ha cumplido de modo que la transición se encontrará activa, moviendo la marca *Fi1c* al lugar Com_{22}, indicando que el fallo se ha propagado.

La recuperación del fallo se realiza de forma similar a la propagación, activándose la $Tcom_{12}$ al desaparacer la marca *Fi1c* del lugar PVf_1 del diagnosticador 1, moviendo la marca *Fi1c* del lugar Com_{12} al lugar Com_{1r}.

Con la marca *Fi1c* en el lugar Com_{1r}, se activa la transición temporizada $Tcom_{1r}$, definida con el tiempo estimado de recuperación. Cuando pase este tiempo, se moverá la marca *Fi1c* al lugar Com_{11}.

Al encontrarse la marca *Fi1c* en el lugar Com_{11}, se activa la transición $Tcom_{21}$, realizándose la comunicación entre los dos sistemas, que moverá la marca *Fi1c* del lugar Com_{22} al lugar Com_{2r}.

Cuando se encuentre la marca *Fi1c* en el lugar Com_{2r}, se activará la transición temporizada $Tcom_{2r}$, estando definida por el tiempo estimado de recuperación, este tiempo ya se ha cumplido por tanto la transición se encontrará activa, y moverá la marca *Fi1c* al lugar Com_{21}, quedando el sistema recuperado.

7.6. Conclusiones del método

Con este método se ha solucionado el problema del tiempo de propagación y recuperación del fallo que se produce con el método de Anidamiento Latente de Fallos Distribuido (Capítulo 6), ofreciendo así una mayor capacidad de diagnóstico.

El método tiene la característica que se puede asignar a cada fallo un tiempo diferente, haciendo un diagnóstico más preciso para cada uno de ellos, ya que no todos los fallos tienen el mismo tiempo de propagación/recuperación.

También se ha conseguido reducir el tiempo en el que el diagnosticador que recibe el fallo dependa del anterior solamente el tiempo necesario, no estando activo el tiempo en el que el fallo se está propagando, mejorando de este modo el diagnóstico.

El inconveniente que presenta el método es la asignación manual del tiempo de propagación/recuperación a cada fallo, que requiere identificar primero todos los fallos que se vayan a comunicar y después el tiempo de propagación/recuperación para cada uno de ellos, habiéndose de realizar este paso en la puesta en marcha del proceso. Éste inconveniente se puede evitar realizando una detección de fallo y asignación de tiempo de forma automática, siendo ésta una de las posibles mejoras al método.

Una forma de obtener los diferentes fallos y el tiempo de propagación/recuperación sería aplicar primeramente el método presentado en el Capítulo 6, y una vez esté aplicado, realizar un autoaprendizaje de los mismos, no siendo de esta forma necesario la ayuda de un experto ni la definición manual de los fallos ni tiempo de propagación/recuperación durante la puesta en marcha, ya que se realizaría de forma automática.

Capítulo 8

Caso práctico

8.1. Descripción del proceso

El caso práctico a estudiar consiste en dos sistemas distribuidos, por una parte está la fabricación de sacos de arena y por otra el paletizado automático de los mismos.

El primer sistema estará compuesto por dos sistemas independientes para tener la posibilidad de fabricar dos tipos de sacos de diferente o igual material (arena, grava,...). El segundo sistema también estará formado por dos sistemas independientes para así poder paletizar los dos tipos de sacos.

El funcionamiento del sistema de forma resumida es el siguiente.

La fabricación de sacos estará formada por una cinta transportadora que llevará el material a una tolva, la cual dispone de un sensor que indica que está llena, parando así la cinta transportadora para dejar de introducir material y abrir una trampilla que dejará caer el material en un tubo.

El tubo estará rodeado por el material del que será el saco (normalmente plástico). Para la confección de dicho saco, habrá un sellador vertical para sellar el plástico y que éste tenga forma circular, y otro sellador horizontal para “cerrar” el saco, que a la vez cortará.

Una vez se ha abierto la trampilla, a los pocos segundos se cerrará y se volverá a activar la cinta de material, a la vez que se activan los selladores, quedando así terminado el saco.

El saco caerá a un sistema de cintas transportadoras que llevarán el saco al sistema de paletizado.

En la cinta transportadora por la que entran los sacos al paletizado habrá un mecanismo para girar los sacos 90º en los casos que corresponda (fila impar) para así tener una buena colocación de los mismos.

El palet estará formado por 9 filas de 6 sacos cada una, los sacos estarán ordenados una fila en un sentido y otra girados 90º.

Un sensor detectará el saco antes de entrar al paletizado para contabilizarlo y saber el número de sacos que hay, para saber si se tiene que activar el actuador para girar los sacos.

Los sacos pasarán a un espacio, que una vez lleno con los 6 sacos, activará dos actuadores que presionarán los sacos y abrirá una compuerta que depositará los sacos en el palet, bajando el palet con los nuevos sacos y cerrando de nuevo las compuertas, para así poder empezar el proceso de nuevo.

Una vez lleno el palet, lo bajará hasta una cinta transportadora que se activará para sacar el palet y volver a introducir uno nuevo, siguiendo con el proceso.

8.2. Modelo inicial sin fallos

Primeramente se va a describir el funcionamiento del sistema sin fallos para tener una mejor comprensión del mismo, mostrando en la Figura 8.1 y en la Figura 8.2 las RdPC de ambos sistemas, pudiendose observar el marcado inicial.

A continuación se muestran las transiciones para el primer sistema, fabricación de sacos de arena.

TR1 =[Cinta 1, Cinta 2]
TR2 =[$\overline{Cinta1}$*,* $\overline{Cinta2}$*, Trampilla 1, Trampilla 2]*
TR3 =[$\overline{Trampilla1}$*,* $\overline{Trampilla2}$*, Sensor V1, Sensor V2, Sensor H1, Sensor H2]*
TR4 =[$\overline{SensorV1}$*,* $\overline{SensorV2}$*,* $\overline{SensorH1}$*,* $\overline{SensorH2}$*]*

Seguidamente se muestran las transiciones para el segundo sistema, paletizado automático.

TR1 =[Cinta 6, Cinta 7, Actuador 5, Cinta 5, Gira Sacos]
TR2 =[$\overline{Cinta6}$, $\overline{Cinta7}$, $\overline{Actuador5}$, $\overline{Cinta5}$, $\overline{GiraSacos}$, Subir Palet 1, Subir Palet 2]
TR3 =[$\overline{SubirPalet1}$·Bajar Palet 1, $\overline{SubirPalet2}$·Bajar Palet 2]
TR4 =[$\overline{BajarPalet1}$, $\overline{BajarPalet2}$]
TR5 =[Compuerta 1, Compuerta 2]
TR6 =[Aprieta Sacos 1, Aprieta Sacos 2, Actuador 1, Actuador 2]
TR7 =[$\overline{Compuerta1}$, $\overline{Compuerta2}$]
TR8 =[$\overline{Compuerta1}$·Bajar Palet 1, $\overline{Compuerta2}$·Bajar Palet 2]
TR9 =[Cinta 6, Cinta7]
TR10 =[$\overline{Cinta6}$, $\overline{Cinta7}$]
TR11 =[$\overline{Actuador1}$, $\overline{Actuador2}$]

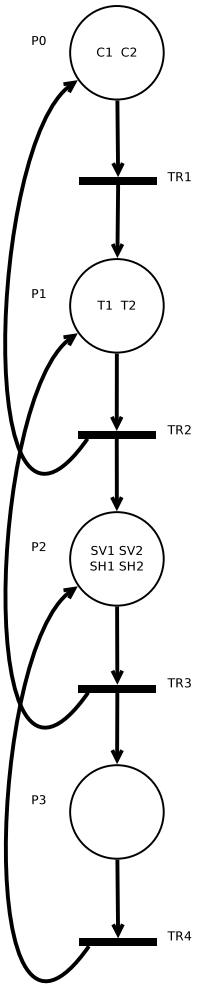


Figura 8.1: RdPC de la fabricación de sacos

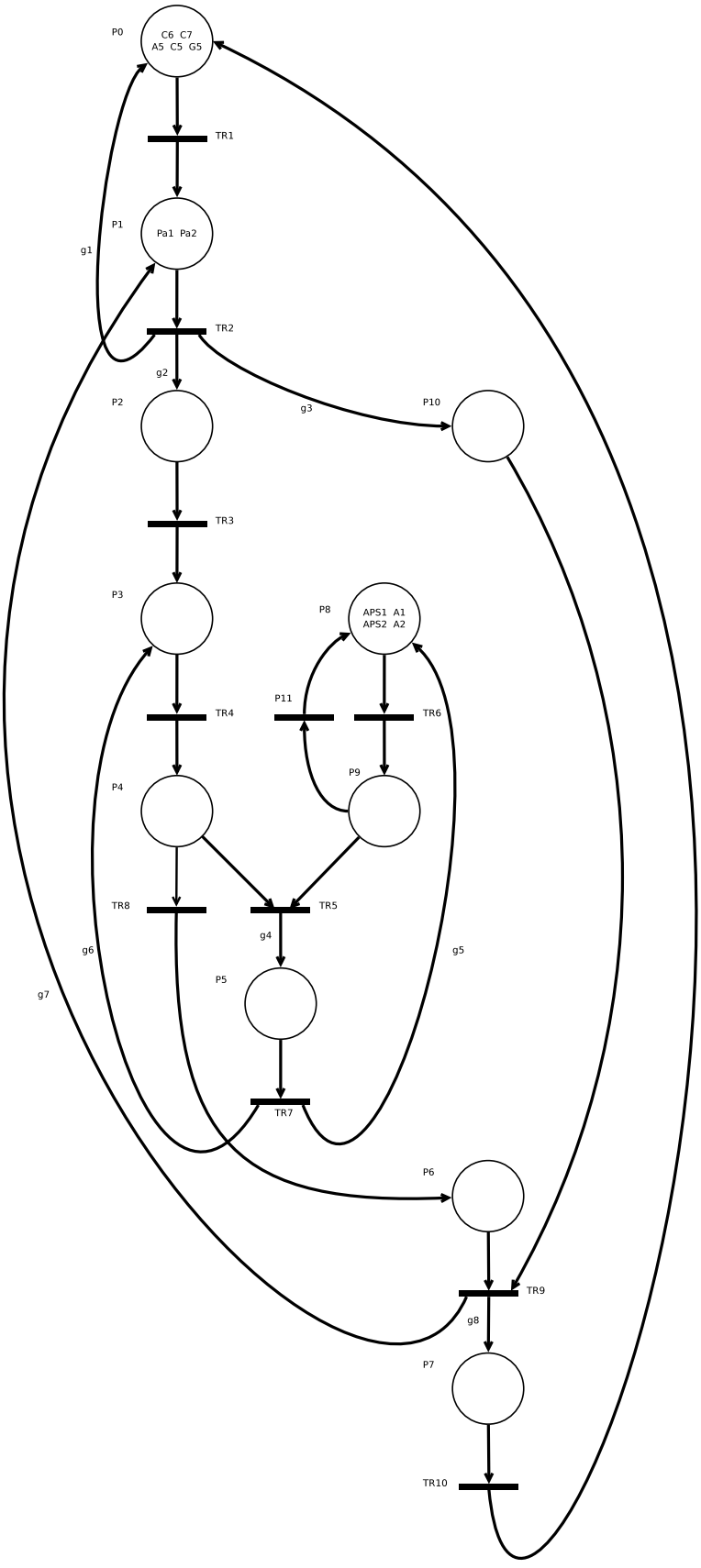


Figura 8.2: RdPC del paletizado automático

En la RdPC del paletizado automático (Figura 8.2) se pueden encontrar funciones de transformación en alguno de los arcos, mostrandose a continuación.

g1(TR2) =[0, 0, A5, C5, G5, 0, 0]
g2(TR2) =[0, 0, 0, 0, 0, Pa1, Pa2]
g3(TR2) =[C6, C7, 0, 0, 0, 0, 0]
g4(TR5) =[Co1, Co2]
g5(TR7) =[APS1, APS2, A1, A2]
g6(TR7) =[Pa1, Pa2]
g7(TR9) =[Pa1, Pa2]
g8(TR9) =[C6, C7]

8.3. Descripción de los fallos

Para determinar los fallos a identificar, primero hay que definirlos manualmente o mediante una base de datos ya creada (si los fallos ya son conocidos).

A continuación se muestran algunos de los fallos del primer sistema. Solamente se muestran algunos fallos para la demostración del método, en futuros estudios se plantea evaluar nuevos fallos.

f1 = la cinta 1 no funciona
f2 = trampilla 1 bloqueada en cerrado
f3 = sellador vertical 1 no sale
f4 = sellador horizontal 1 no sale

Seguidamente se muestran algunos fallos del segundo sistema, se evaluarán nuevos fallos en futuros estudios.

f5 = la cinta 7 no funciona
f6 = palet 1 no sube
f7 = aprieta sacos 1 no sale
f8 = compuerta 2 no abre

8.4. Trayectorias de verificación y recuperación de fallos

Estas trayectorias son las representaciones que unen el lugar *PLNf* con el lugar *PVf* y viceversa. Se definen por las transiciones de fallo (*Tf*) y de recuperación (*Tr*) respetivamente y dependerán de las medidas de los sensores.

Los sensores de los que se dispone para el primer sistema son:

$srov_1(M_k) = \{On_1, Off_1\} \Rightarrow$ *Velocidad*
$srov_2(M_k) = \{On_2, Off_2\} \Rightarrow$ *Presencia*
$srov_3(M_k) = \{On_3, Off_3\} \Rightarrow$ *Presencia*
$srov_4(M_k) = \{On_4, Off_4\} \Rightarrow$ *Presencia*

Para el segundo sistema se dispone de los siguientes sensores:

$srov_5(M_k) = \{On_5, Off_5\} \Rightarrow$ *Velocidad*
$srov_6(M_k) = \{On_6, Off_6\} \Rightarrow$ *Velocidad*
$srov_7(M_k) = \{On_7, Off_7\} \Rightarrow$ *Presencia*
$srov_8(M_k) = \{On_8, Off_8\} \Rightarrow$ *Presencia*

Donde *Velocidad* es un sensor de movimiento, que detecta si está en funcionamiento o está parado, *Presencia* es un sensor que detecta si el cilindro se ha movido, teniendo los dos sensores salidas de tipo binario.

A continuación se muestran los valores de las lecturas sensoriales de verificación y de recuperación de fallo para cada $PLNf$ de cada fallo a diagnosticar.

Para el primer sistema:

- *Cinta 1:*

Lugar	$PLNf$	f_1
P1	Off_1	Off_1
P2	On_1	Off_1

- *Trampilla 1:*

Lugar	$PLNf$	f_2
P2	Off_2	Off_2
P3	On_2	Off_2

- *Sellador Vertical 1:*

Lugar	$PLNf$	f_3
P3	Off_3	Off_3
P4	On_3	Off_3

- *Sellador Horizontal 1:*

Lugar	$PLNf$	f_4
P3	Off_4	Off_4
P4	On_4	Off_4

Para el segundo sistema:

- *Cinta 7:*

Lugar	$PLNf$	f_5
P1	Off_5	Off_5
P2	On_5	Off_5
P11	Off_5	Off_5
P8	On_5	Off_5

- *Subir palet 1:*

Lugar	$PLNf$	f_6
P2	Off_6	Off_6
P3	On_6	Off_6

- *Aprieta sacos 1:*

Lugar	$PLNf$	f_7
P9	Off_7	Off_7
P10	On_7	Off_7

- *Compuerta 2:*

Lugar	$PLNf$	f_8
P6	On_8	Off_8

A continuación se van a definir las diferentes transiciones de fallo.

Para el primer sistema:

TF1 =$[srov_{nev}(f_1)]$
TF2 =$[srov_{ev}(f_1)]$
TF3 =$[srov_{nev}(f_2)]$
TF4 =$[srov_{ev}(f_2)]$
TF5 =$[srov_{nev}(f_3), srov_{nev}(f_4)]$
TF6 =$[srov_{ev}(f_3), srov_{ev}(f_4)]$

Para el segundo sistema:

TF1 =$[srov_{nev}(f_5)]$
TF2 =$[srov_{ev}(f_5)]$
TF3 =$[srov_{nev}(f_6)]$
TF4 =$[srov_{ev}(f_6)]$
TF5 =$[srov_{nev}(f_7)]$
TF6 =$[srov_{ev}(f_7)]$
TF7 =$[srov_{nev}(f_8)]$
TF8 =$[srov_{ev}(f_8)]$
TF9 =$[srov_{nev}(f_5)]$
TF10 =$[srov_{ev}(f_5)]$

8.5. Lugares de anidamiento latente de fallos

Los lugares de anidamiento latente de fallos son los lugares de la RdPC en los que se encuentran las marcas de fallo.

Para saber los lugares en los que colocar las marcas de fallo, hemos de buscar en las tablas de las lecturas sensoriales la diferencia entre la lectura sensorial normal y la de fallo.

En la Figura 8.3 y la Figura 8.4 se pueden ver las RdPCDF de los dos sistemas.

8.6. Definición del conjunto de fallos y tiempo de propagación/recuperación

En la Sección 8.3 se han definido los fallos del sistema, pero para el método propuesto se han de definir los fallos que se van a transmitir junto con el tiempo de propagación y recuperación de cada uno de ellos para añadirlos a los lugares de comunicación de los diferentes subsistemas.

A continuación se definen algunos de los fallos del sistema de fabricación de sacos que se van a comunicar al sistema de paletizado y sus tiempos de propagación/recuperación (tp/tr).

f3 = sellador vertical 1 no sale	*tp3 = 5 s. tr3 = 7 s.*
f4 = sellador horizontal 1 no sale	*tp4 = 5 s. tr4 = 7 s.*

Los tiempos de propagación/recuperación se han definido mediante diferentes pruebas realizadas durante la implementación del método, considerando este tiempo suficiente para la detección del fallo con un margen de seguridad. Se ha obtenido un tiempo medio de propagación real del fallo de 6 segundos.

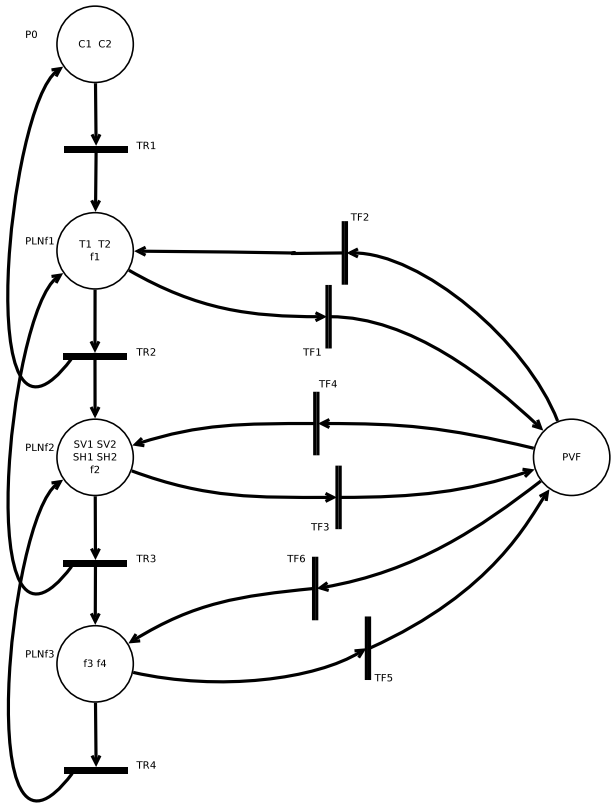


Figura 8.3: RdPCDF de la fabricación de sacos

En el caso a estudiar solamente se ha tenido en cuenta la comunicación de fallos del sistema de fabricación de sacos al sistema de paletizado. Éstos fallos solamente van a afectar al *f7* (aprieta sacos 1 no sale) del sistema de paletizado.

8.7. Modificaciónes en los diagnosticadores

Las modificaciones consisten en añadir una función *Post* a la transición de fallo y recuperación de los diferentes diagnosticadores para que modifiquen las marcas según las condiciones de disparo de cada transición.

A continuación se definen las funciones de los arcos del sistema de paletizado automático.

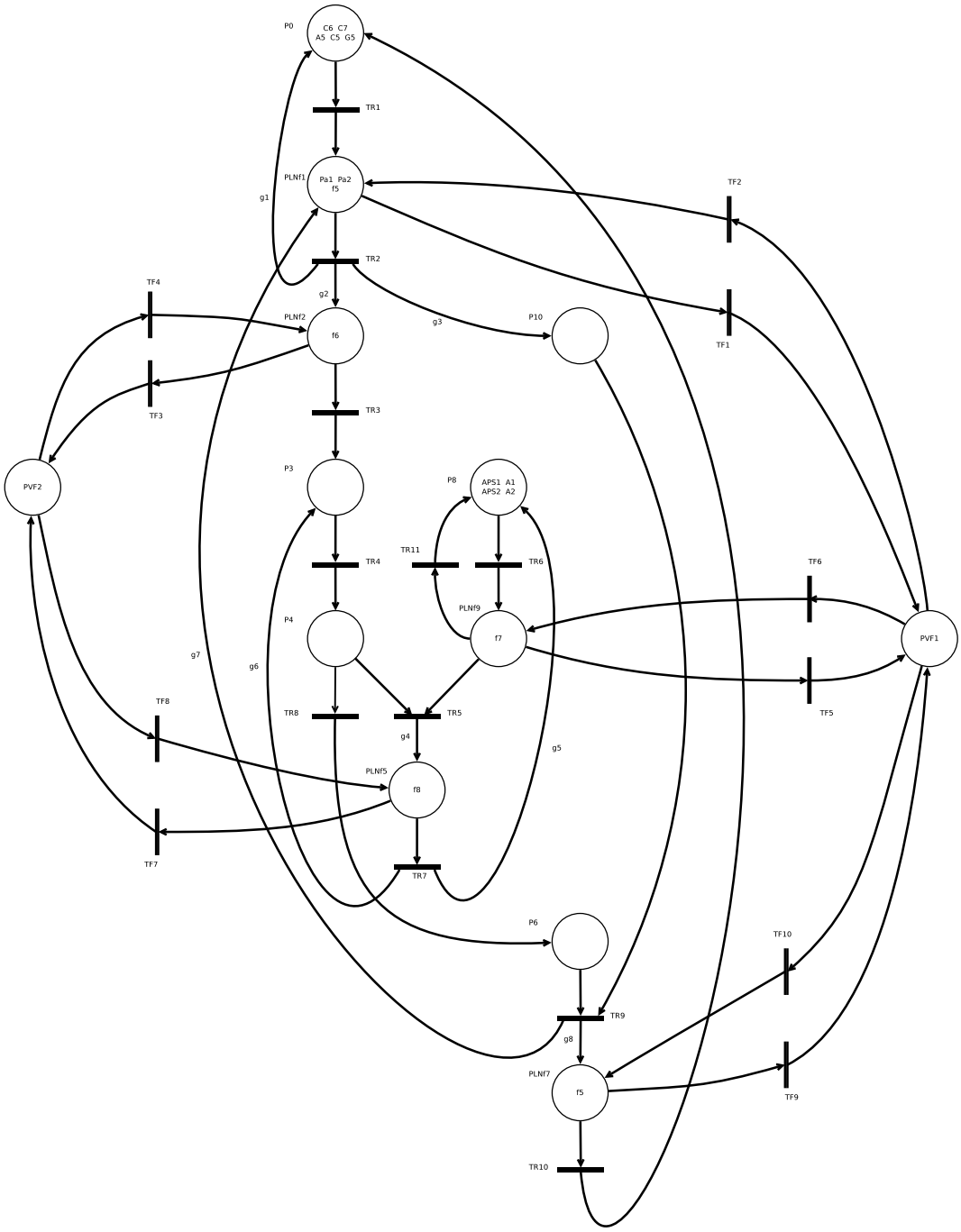


Figura 8.4: RdPCDF de la fabricación de sacos

Definición 22 *Las funciones de los arcos se definen:*

$$Post(TF5, PLNf_3/f7) = \begin{cases} f7 & si \quad M(Com_{22}(< 0 >)) \\ f7,3 & si \quad M(Com_{22}(< f3 >)) \\ f7,4 & si \quad M(Com_{22}(< f4 >)) \end{cases}$$

$$Post(PVf_1, TF6) =< f7,3 > \cup < f7,4 > \cup < f7 >= f7$$

8.8. Trayectorias de comunicación de fallos

Para realizar la comunicación entre los diagnosticadores y los lugares de comunicación, y viceversa, se utilizan las trayectorias de comunicación de fallos, activándose éstas con la aparición de marcas en los lugares de comunicación y de los diagnosticadores.

Las trayectorias de fallos de los lugares de comunicación del sistema de fabricación de fallos son las siguientes.

- Trayectoria de fallo del lugar de comunicación que detecta el fallo

$$M(Com_{11}(< f3 >, < f4 >))$$
$$[Tcom_{11}/SROV_{nev}(M(Com_{11}(< f3 >, < f4 >)))$$
$$> M(Com_{1p}(< f3 >, < f4 >))$$

  Donde $Tcom_{11}$ es la transición que se activa al aparecer la marca $f3$ o la marca $f4$ en el lugar PVf del diagnosticador de fabricación de sacos.

- Trayectoria de fallo del lugar de comunicación que propaga el fallo

$$M(Com_{1p}(< f3 >, < f4 >))$$
$$[Tcom_{1p}/SROV_{nev}(M(Com_{1p}(< f3 >, < f4 >)))$$
$$> M(Com_{12}(< f3 >, < f4 >))$$

  Donde $Tcom_{1p}$ es la transición temporizada que se activa al aparecer la marca $f3$ o la marca $f4$ en el lugar Com_{1p}, donde el tiempo se define como tiempo de propagación diferente para cada uno de los fallos.

– Trayectoria de recuperación del lugar de comunicación que detecta el fallo

$$M(Com_{12}(< f3 >, < f4 >))$$

$$[Tcom_{12}/SROV_{ev}(M(Com_{12}(< f3 >, < f4 >)))$$

$$> M(Com_{1r}(< f3 >, < f4 >))$$

Donde $Tcom_{12}$ es la transición que se activa al desaparecer la marca $f3$ o la marca $f4$ del lugar PVf del diagnosticador de fabricación de sacos.

– Trayectoria de recuperación del lugar de comunicación que recupera el fallo

$$M(Com_{1r}(< f3 >, < f4 >))$$

$$[Tcom_{1r}/SROV_{nev}(M(Com_{1r}(< f3 >, < f4 >)))$$

$$> M(Com_{11}(< f3 >, < f4 >))$$

Donde $Tcom_{1r}$ es la transición temporizada que se activa al aparecer la marca $f3$ o la marca $f4$ en el lugar Com_{1r}, donde el tiempo se define como tiempo de recuperación diferente para cada fallo.

A continuación se definen las trayectorias de fallos de los lugares de comunicación del sistema de paletizado automático, siendo en este paso en el que se realiza la comunicación entre los dos sistemas.

– Trayectoria de fallo del lugar de comunicación que recibe el fallo

$$M(Com_{21}(< f3 >, < f4 >))$$

$$[Tcom_{21}/SROV_{nev}(M(Com_{21}(< f3 >, < f4 >)))$$

$$> M(Com_{2p}(< f3 >, < f4 >))$$

Donde $Tcom_{21}$ es la transición que se activa al aparecer la marca $f3$ o la marca $f4$ en Com_{12}.

– Trayectoria de fallo del lugar de comunicación que propaga el fallo

$$M(Com_{2p}(< f3 >, < f4 >))$$

$$[Tcom_{2p}/SROV_{nev}(M(Com_{2p}(< f3 >, < f4 >)))$$

$$> M(Com_{22}(< f3 >, < f4 >))$$

Donde $Tcom_{2p}$ es la transición temporizada que se activa al aparecer la marca $f3$ o la marca $f4$ en el lugar Com_{2p}, estando activa la transición debido a que el tiempo de propagación ya se ha cumplido.

– Trayectoria de recuperación del lugar de comunicación que recibe el fallo

$$M(Com22(< f3 >, < f4 >))$$

$$[Tcom_{22}/SROV_{ev}(M(Com_{22}(< f3 >, < f4 >)))$$

$$> M(Com_{2r}(< f3 >, < f4 >))$$

Donde $Tcom_{22}$ es la transición que se activa al aparecer la marca $f3$ o la marca $f4$ en Com_{11}.

– Trayectoria de recuperación del lugar de comunicación que recupera el fallo

$$M(Com_{2r}(< f3 >, < f4 >))$$

$$[Tcom_{2r}/SROV_{nev}(M(Com_{2r}(< f3 >, < f4 >)))$$

$$> M(Com_{21}(< f3 >, < f4 >))$$

Donde $Tcom_{2r}$ es la transición temporizada que se activa al aparecer la marca $f3$ o la marca $f4$ en el lugar Com_{2r}, al haberse cumplido el tiempo de recuperación, ésta ya se encontrará activada.

8.9. Implementación

Para la implementación de este método se ha utilizado el diagrama que se muestra en la Figura 8.5, empleando *Labview* para las labores de diagnóstico, monitorización y comunicación de los fallos mediante un sistema Scada, utilizando una arquitectura cliente-servidor con un protocolo *TCP-IP* para la comunicación entre diagnosticadores. El control lo realizan dos autómatas *Omron CQM1H*, conectados entre sí mediante *Controller-link*. Los diagnosticadores se comunicarán con los autómatas a través de una conexión serie *RS-232*.

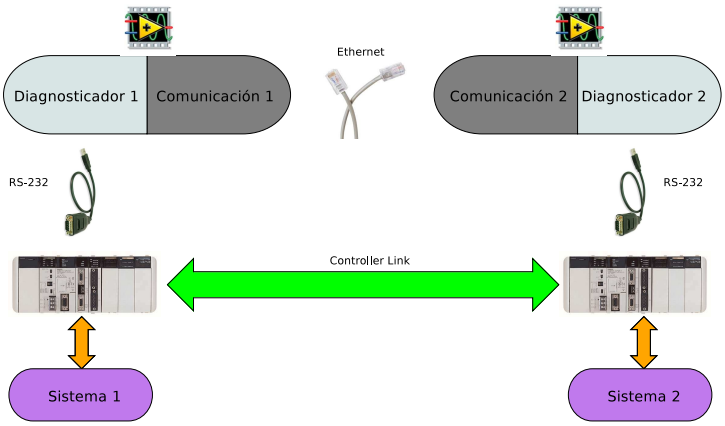


Figura 8.5: Diagrama de implementación

El sistema Scada está compuesto por una RdPCDFDR que realiza la monitorización y diagnóstico del funcionamiento del sistema y de los fallos de este, una RdPC que realiza la monitorización de la comunicación de los fallos, un cliente que envía el estado del sistema y los mensajes de fallo, y un servidor que recibe el estado de los demás sistemas.

En la Figura 8.6 se muestra el ejemplo de como queda el sistema de fabricación de sacos, donde se puede ver la RdPCDFDR del sistema, la RdPC de comunicación, el estado de las transiciones, las entradas del sistema y el modo de conexión (Local/PLC), así como el marcado inicial y el actual. Se puede ver también que se ha producido el fallo *f3* (Sellador Vertical 1 no sale), y se ha comunicado al otro sistema después de haber transcurrido el tiempo estimado de propagación, esto último se deduce al encontrarse la marca de fallo en el lugar de comunicación *PC3*.

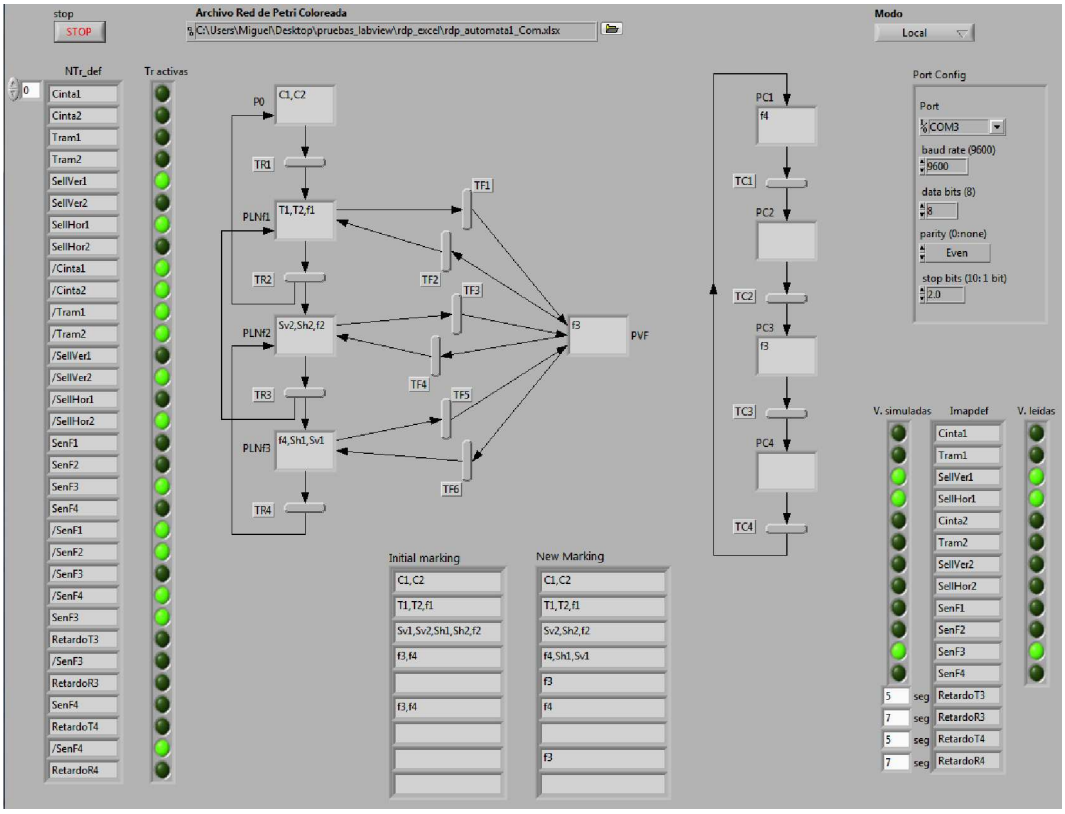


Figura 8.6: Captura de pantalla del Scada sistema de fabricación de sacos

A continuación, en la Figura 8.7, se muestra como se representaría en el Scada del sistema de paletizado que se ha producido el fallo *f4* (Aprieta Sacos 1 no sale) como consecuencia del fallo producido en el sistema de fabricación de sacos *f3*. Se puede apreciar como en el lugar de comunicación *PC3* se ha recibido el fallo *f3* del sistema de fabricación de sacos, indicando que el fallo ya se ha propagado hasta este sistema.

Para la obtención del tiempo de propagación y recuperación del fallo, se han realizado diferentes pruebas con el sistema de control, para medir el tiempo real que tarda el fallo en propagarse y en recuperarse, obteniendo un tiempo medio de 6 segundos, tomando como

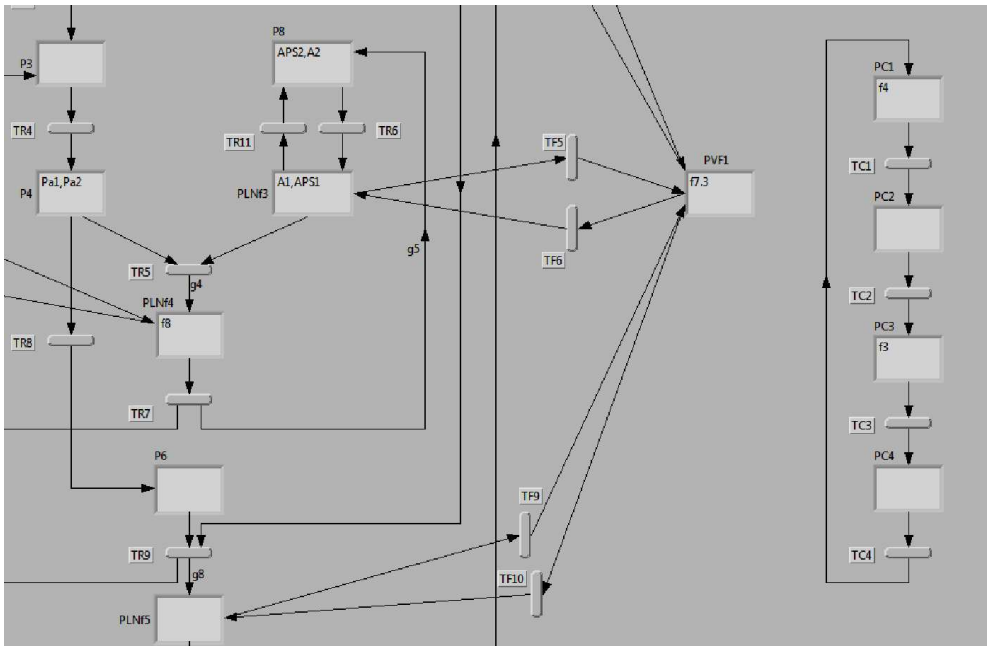


Figura 8.7: Captura de pantalla del Scada sistema de paletizado

tiempo estimado de propagación 5 segundos y como tiempo estimado de recuperación 7 segundos, considerando un margen de seguridad suficiente para la detección del fallo.

Figura 8.8: Tiempos estimados de propagación y recuperación

A continuación se muestran diferentes ensayos realizados en el laboratorio para comprobar el funcionamiento del método.

Ensayo 1

El ensayo 1 se realiza para comprobar el correcto funcionamiento del sistema de control así como la correcta monitorización del sistema mediante la RdPC. En este ensayo no se produce ningún tipo de fallo, solamente se realizan diferentes pruebas de funcionamiento normal del sistema.

Ensayo 2

En este ensayo se van a provocar diferentes fallos de forma individualizada, con el fin de verificar la detección de cada fallo por separado y garantizar su aislamiento en el lugar de anidamiento, verificando el correcto funcionamiento del Método de Anidamiento Latente de Fallos en sistemas distribuidos.

Se han probado de forma individual los fallos: *f1* (Cinta 1 no funciona), *f2* (Trampilla 1 bloqueada en cerrado), *f5* (Cinta 7 no funciona), *f6* (Palet 1 no sube) y *f8* (Compuerta 2 no abre), comprobando que se hace un correcto diagnóstico y aislamiento de cada uno de ellos.

Ensayo 3

En el siguiente ensayo, al igual que en el anterior, se comprueba el correcto funcionamiento del Método de Anidamiento Latente de Fallos aplicado a sistemas SED's complejos distribuidos, realizando pruebas con la detección de fallos múltiples en los que no se produce acoplamiento entre los sistemas, es decir, los fallos producidos no se van a propagar al siguiente sistema.

Se han probado los mismos fallos que en el ensayo anterior, en este caso de forma simultánea, comprobando de nuevo la potencia de diagnóstico que ofrece el método de Anidamiento Latente de Fallos.

Ensayo 4

En el cuarto ensayo se comprueban las mejoras presentadas del Método de Anidamiento Latente de Fallos Distribuido con Retardo, provocando fallos que producen acoplamiento entre los sistemas de forma que se comprueba la comunicación entre los diagnosticadores y la mejora en el diagnóstico al introducir el tiempo estimado de propagación y el tiempo estimado de recuperación.

En este ensayo se han realizado dos pruebas, la primera prueba consiste en la detección del fallo por el segundo sistema una vez se ha producido un fallo en el primero y ha transcurrido el tiempo estimado de propagación, produciéndose el fallo en el segundo sistema como consecuencia del fallo del primero. En la Figura 8.6 y Figura 8.7 se muestra uno de los ensayos realizados para esta primera prueba. La segunda prueba es la detección del fallo del segundo sistema antes de que haya transcurrido el tiempo estimado de propagación del fallo producido en el primer sistema, detectando así el fallo como propio, ya que el fallo del primer sistema todavía no se ha propagado.

Capítulo 9

Conclusiones y trabajo futuro

Con esta tesina se presenta una nueva técnica para el diagnóstico de fallos en SED's complejos distribuidos utilizando el método de Anidamiento Latente de Fallos, mejorando también la monitorización de los mismos.

La primera aportación de esta tesina muestra la ampliación del método de Anidamiento Latente de Fallos para aplicarlo en SED's complejos distribuidos. Con ésto se ha conseguido evitar que se realice un diagnóstico erróneo del fallo producido debido al flujo compartido mediante la comunicación entre los diagnosticadores, observando que se pierde poder de diagnóstico ya que al realizar la comunicación, el diagnosticador que recibe el fallo no puede diagnosticar un fallo como propio mientras el otro se encuentre activo.

La segunda aportación es la mejora a la técnica presentada, desarrollando el método de Anidamiento Latente de Fallos Distribuido con Retardo. En él, como dice el nombre, se tienen en cuenta los retardos de propagación y recuperación de fallos que se producen en los sistemas distribuidos, consiguiendo que el diagnosticador que recibe el fallo se encuentre detectando el fallo del otro sistema solamente el tiempo necesario, mejorando así el poder de diagnóstico.

En la Figura 9.1 se puede observar el diagrama de tiempos de la propagación de fallos, donde se puede ver como la detección de fallo se realiza en un instante, a continuación se envía el fallo una vez pasado un tiempo estimado de propagación y por último el diagnosticador que recibe el fallo lo detecta cuando ha transcurrido el tiempo de propagación real del fallo. Se ha conseguido reducir el tiempo en que el diagnosticador que recibe el fallo detecta el fallo del otro sistema solamente al tiempo de espera de fallo, pudiéndose ajustar éste en función del tiempo estimado de propagación, dependiendo de la precisión que se quiera en el diagnóstico de cada fallo.

En la Figura 9.2 se encuentra el diagrama de tiempos de recuperación de fallos, donde está el diagrama de recuperación del fallo, el envío de la recuperación y la recuperación real del diagnosticador que recibe el fallo. Se observa como en este caso el tiempo estimado de re-

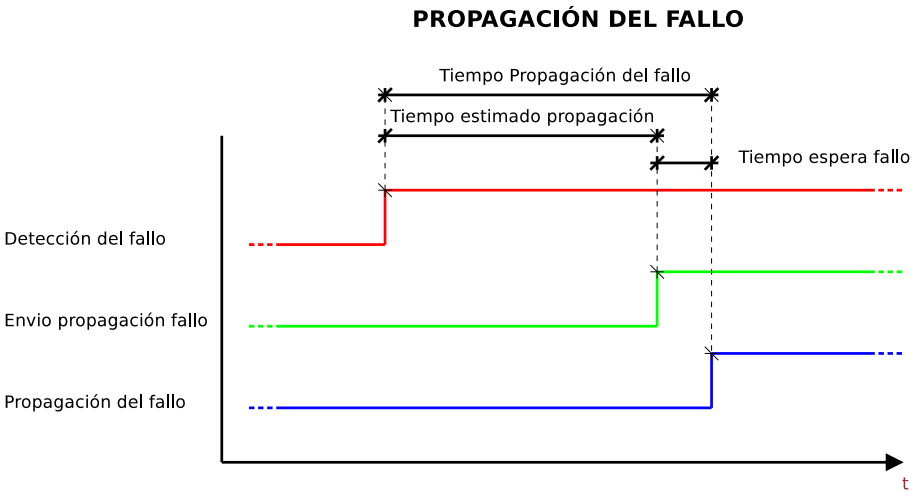


Figura 9.1: Diagrama de tiempos de la propagación de fallos

cuperación es mayor que el tiempo real de recuperación del fallo, esto es debido a que el diagnosticador debe estar detectando el fallo hasta que se haya recuperado el diagnosticador para evitar de este modo un diagnóstico erróneo, pudiendo ajustar el tiempo de detección de fallo según la precisión de diagnóstico deseada modificando el tiempo estimado de recuperación.

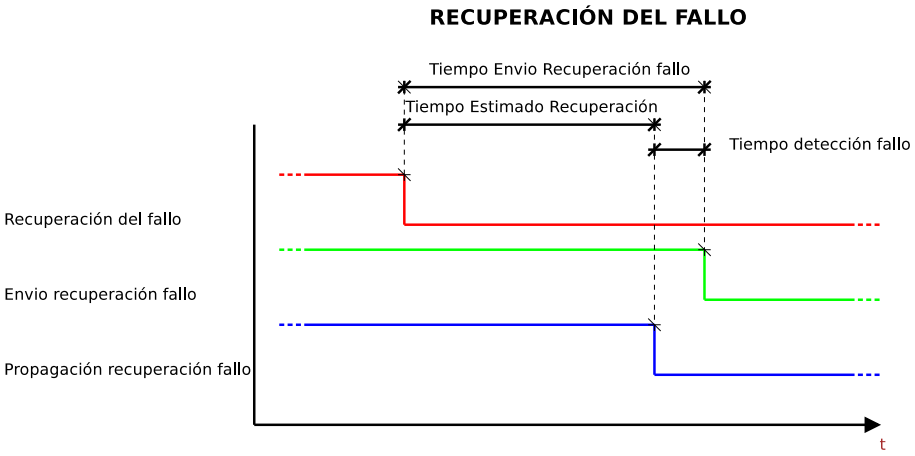


Figura 9.2: Diagrama de tiempos de la recuperación de fallos

Con esta nueva técnica se ha conseguido reducir el tiempo en que el diagnosticador que recibe el fallo esté detectando el fallo del otro sistema, ajustándose éste modificando el tiempo estimado de propagación y el tiempo estimado de recuperación, consiguiendo de esta forma un diagnóstico de fallo más preciso así como mayor poder de diagnóstico. También se ha independizado el sistema de comunicación entre los diagnosticadores y el sistema de comunicación del sistema de control, evitando así posibles problemas debidos a la comunicación.

Un inconveniente que presenta este método es la asignación de los fallos y del tiempo de propagación/recuperación de los mismos, habiendo de identificar cada uno de ellos manual-

mente. Este inconveniente se plantea como un trabajo futuro de cara a desarrollar nuevos métodos de autodetección de fallos.

Una ampliación a desarrollar en trabajos futuros sería ampliar el método para su uso no solo en sistemas discretos, sino también en sistemas contínuos e híbridos, abarcando así un mayor número de sistemas a diagnosticar, debido a que muchos procesos están controlados a la vez por variables discretas y continuas.

En trabajos futuros también se podrían aplicar técnicas de inteligencia artificial como la lógica difusa, redes neuronales y algunos otros para el modelado del sistema de diagnóstico, así como para la identificación de fallos.

Bibliografía

Aramburo-Lizarraga, J., E. Lopez-Mellado and A. Ramirez-Trevino (2005). Distributed fault diagnosis using petri net reduced models. In: *Systems, Man and Cybernetics, 2005 IEEE International Conference on*. Vol. 1. pp. 702–707 Vol. 1. 2, 17

Berthomieu, B. and M. Diaz (1991). Modeling and verification of time dependent systems using time petri nets. *Software Engineering, IEEE Transactions on* **17**(3), 259–273. 4

Chen, Jie and Ron J. Patton (1999). *Robust model-based fault diagnosis for dynamic systems*. Kluwer Academic Publishers. Norwell, MA, USA. 11

Correcher, A., E. García, F. Morant and E. Quiles (2001). Self-learning coloured petri nets for diagnosing non modeled faults. 17

David, René and Hassane Alla (2005). *Discrete, continuous, and hybrid Petri Nets*. 4

Fabre, E., A. Benveniste and C. Jard (2002). Distributed diagnosis for large discrete event dynamic systems. In: *15th IFAC World Congress, Barcelona*. 2, 17

Florin, G., C. Fraize and S. Natkin (1991). Stochastic petri nets: Properties, applications and tools. *Microelectronics Reliability* **31**(4), 669–697. 4

Garcia, E., F. Morant, R. Blasco-Gimenez, A. Correcher and E. Quiles (2002). Centralized modular diagnosis and the phenomenon of coupling. In: *Discrete Event Systems, 2002. Proceedings. Sixth International Workshop on*. pp. 161–168. 2, 17

Garcia, E., L. Rodriguez, F. Morant, A. Correcher and E. Quiles (2008*a*). Latent nestling method: A new fault diagnosis methodology for complex systems. In: *Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE*. pp. 253–258. I, 31

Garcia, E., L. Rodriguez, F. Morant, A. Correcher, E. Quiles and R. Blasco (2008*b*). Fault diagnosis with coloured petri nets using latent nestling method. In: *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*. pp. 986–991. I, 17, 25

Garcia, Emilio (2000). *Automatización de procesos industriales*. Alfaomega. 7

Genc, S. and S. Lafortune (2007). Distributed diagnosis of Place-Bordered petri nets. *Automation Science and Engineering, IEEE Transactions on* **4**(2), 206–219. 22

Genc, Sahika and Stéphane Lafortune (2003). Distributed diagnosis of discrete-event systems using petri nets.. Springer-Verlag. pp. 316–336. 2, 15, 17, 22

Gertler, Janos J. (1998). *Fault detection and diagnosis in engineering systems*. New York :Marcel Dekker. 11

Hashtrudi Zad, S., R.H. Kwong and W.M. Wonham (2003). Fault diagnosis in discrete-event systems: framework and model reduction. *Automatic Control, IEEE Transactions on* **48**(7), 1199 – 1212. 2

Jensen, Kurt (1995). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use (Monographs in Theoretical Computer Science a Series of Eatcs)*. Springer. 1

Kan, Chieh-Ying and Xudong He (1996). A method for constructing algebraic petri nets. *Journal of Systems and Software* **35**(1), 15–27. 4

Kasturia, E., F. DiCesare and A. Desrochers (1988). Real time control of multilevel manufacturing systems using colored petri nets. In: *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*. pp. 1114–1119 vol.2. 15

Kiliç, Erdal, Çağlar Karasu and Kemal Leblebicioğlu (2006). Fault diagnosis with dynamic fuzzy discrete event system approach. In: *Artificial Intelligence and Neural Networks* (F. Savaci, Ed.). Vol. 3949 of *Lecture Notes in Computer Science*. pp. 117–124. Springer Berlin / Heidelberg. 15

Lin, Feng (1994). Diagnosability of discrete event systems and its applications. *Discrete Event Dynamic Systems* **4**(2), 197–212. 1

Moreno, Emilio Garcia (2000). Descomposición modular de diagnosticadores de fallos basados en modelos de eventos discretos. PhD thesis. Universidad Politécnica de Valencia. 15

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE* **77**(4), 541–580. 1

Pedrycz, W. and F. Gomide (1994). A generalized fuzzy petri net model. *Fuzzy Systems, IEEE Transactions on* **2**(4), 295–301. 4

Petri, CA (1962). Kommunikation mit Automaten. PhD thesis. Institut für instrumentelle Mathematik. 4, 8

Ramadge, P. J. and W. M. Wonham (1987). Modular feedback logic for discrete event systems. *SIAM Journal on Control and Optimization* **25**(5), 1202–1218. 15

Rodríguez, L., E. García, F. Morant, A. Correcher and E. Quiles (2010). Formalización del método de anidamiento latente para el diagnóstico de fallos en sistemas híbridos. *XIV Congreso Latinoamericano de Control Automático, Santiago de Chile*. 17

Rodríguez, L., E. García, F. Morant, A. Correcher, E. Quiles and V. Fluixá (2008*a*). Aplicación del método de anidamiento latente de fallos usando redes de petri coloreadas para el diagnóstico de fallos en el sistema de refrigeración y lubricación de un aerogenerador. *síntesis* **11**, 12. 17, 30

Rodríguez, L., E. García, F. Morant, A. Correcher, E. Quiles and V. Fluixá (2008*b*). Método de anidamiento latente de fallos usando rdpc aplicado a los subsistemas de un aerogenerador. *XIII Congreso Latinoamericano de Control Automático / VI Congreso Venezolano de Automatización y Control, Métida, Venezuela*. 17

Rodríguez, Leonardo (2009). Diagnóstico de fallos en sistemas complejos basado en la metodología de anidamiento latente. caso de estudio: Sistema de refrigeración y lubricación de la multiplicadora de un aerogenerador. Master's thesis. Universidad Politécnica de Valencia. 16

Sampath, M., R. Sengupta, S. Lafortune, K. Sinnamohideen and D. Teneketzis (1995). Diagnosability of discrete-event systems. *Automatic Control, IEEE Transactions on* **40**(9), 1555–1575. 15

Su, R., W.M. Wonham, J. Kurien and X. Koutsoukos (2002). Distributed diagnosis for qualitative systems. In: *Discrete Event Systems, 2002. Proceedings. Sixth International Workshop on*. pp. 169–174. 2, 20

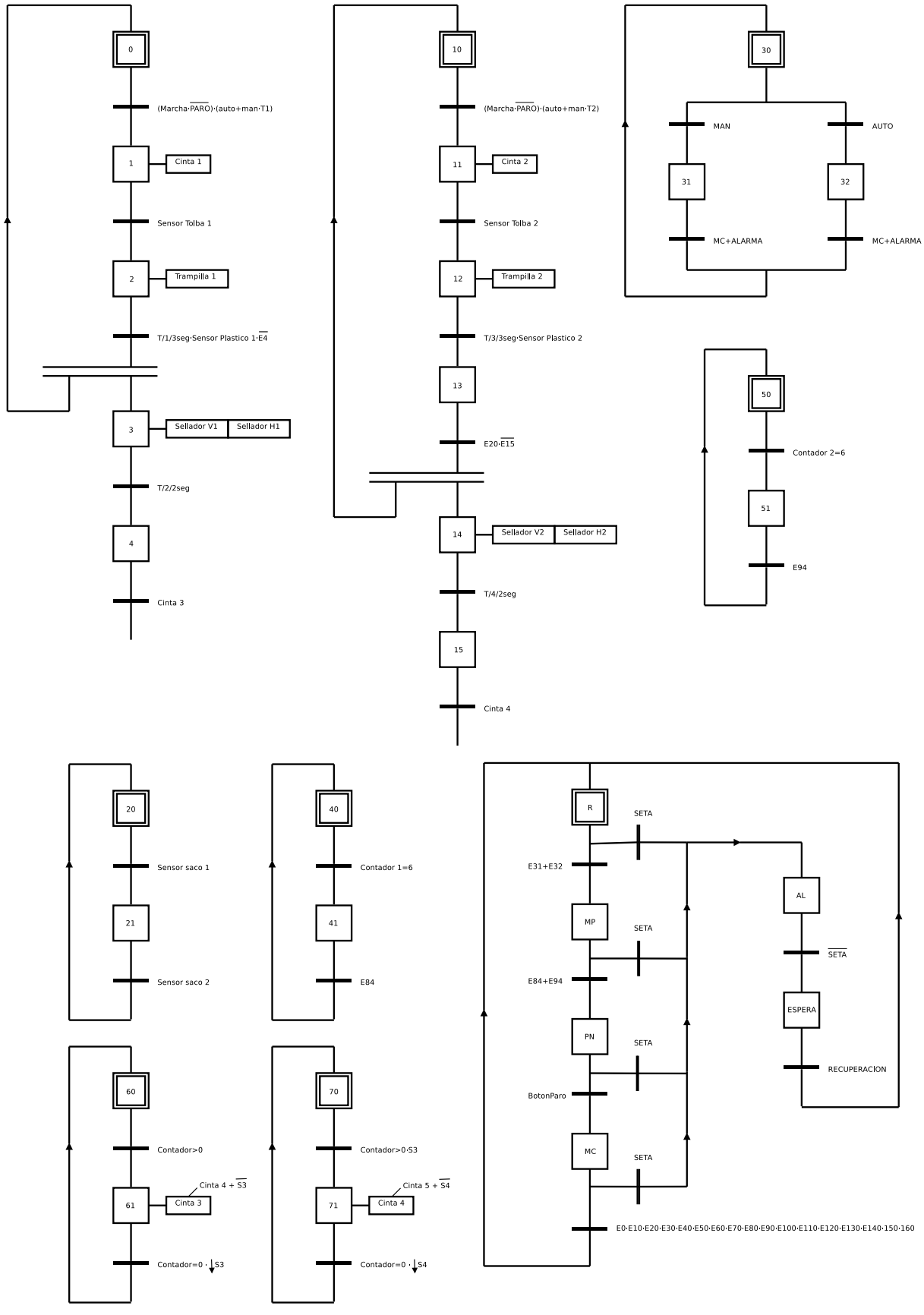
Viswanadham, N. and T.L. Johnson (1988). Fault detection and diagnosis of automated manufacturing systems. In: *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*. pp. 2301–2306 vol.3. 15

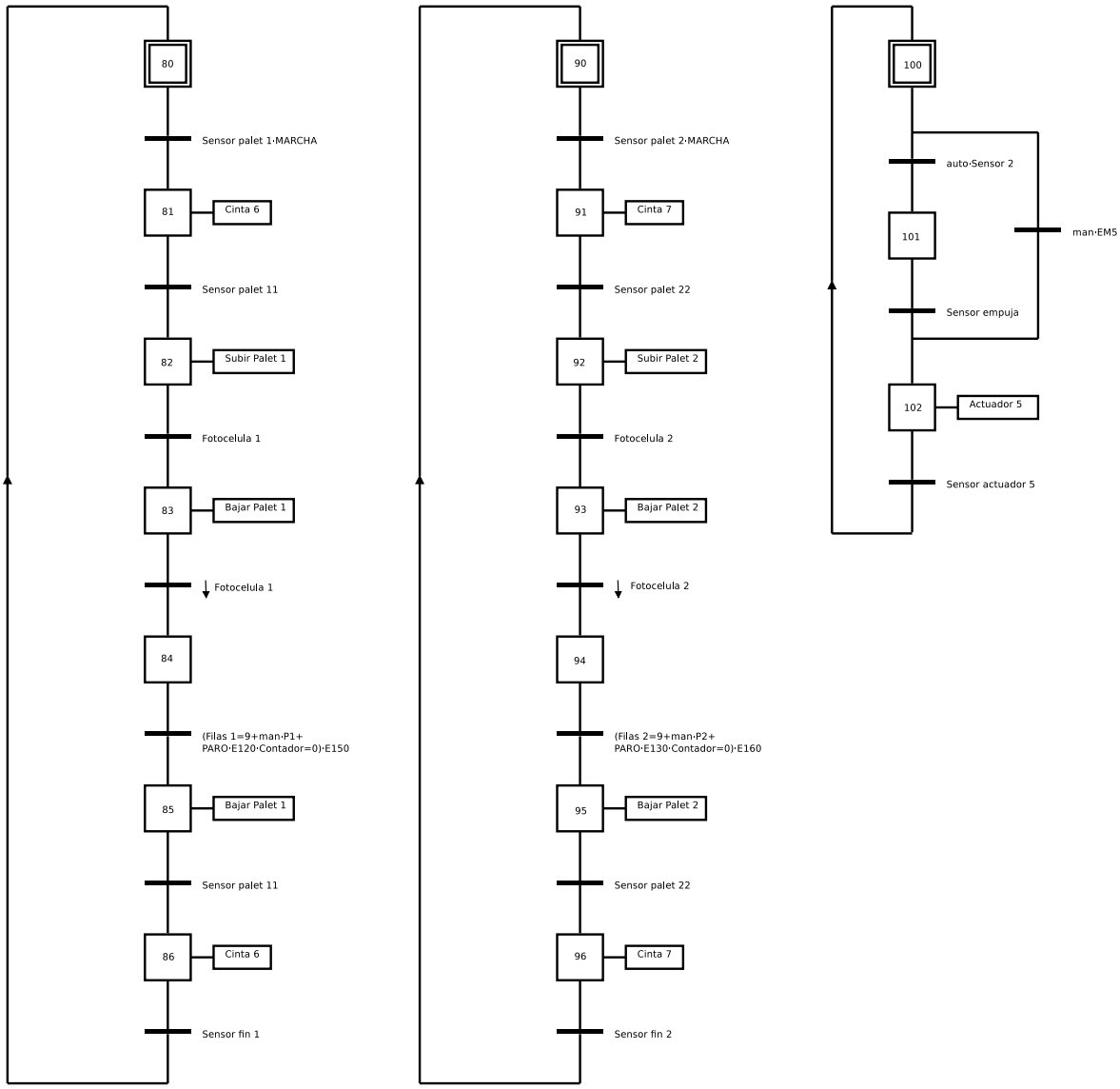
Xue, Fei and Lu Yan (2007). Formal approach to fault diagnosis in distributed discrete event systems with OBDD. *Innovations in Systems and Software Engineering* **3**(4), 259–267. 15

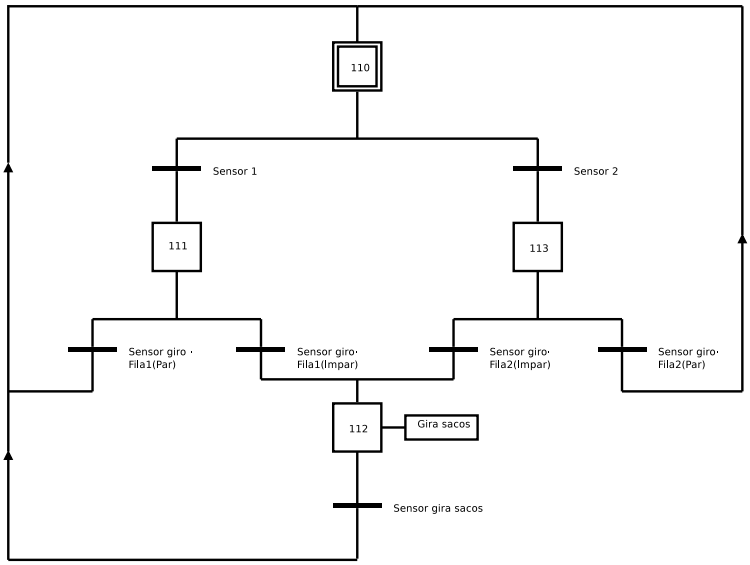
Anexo:

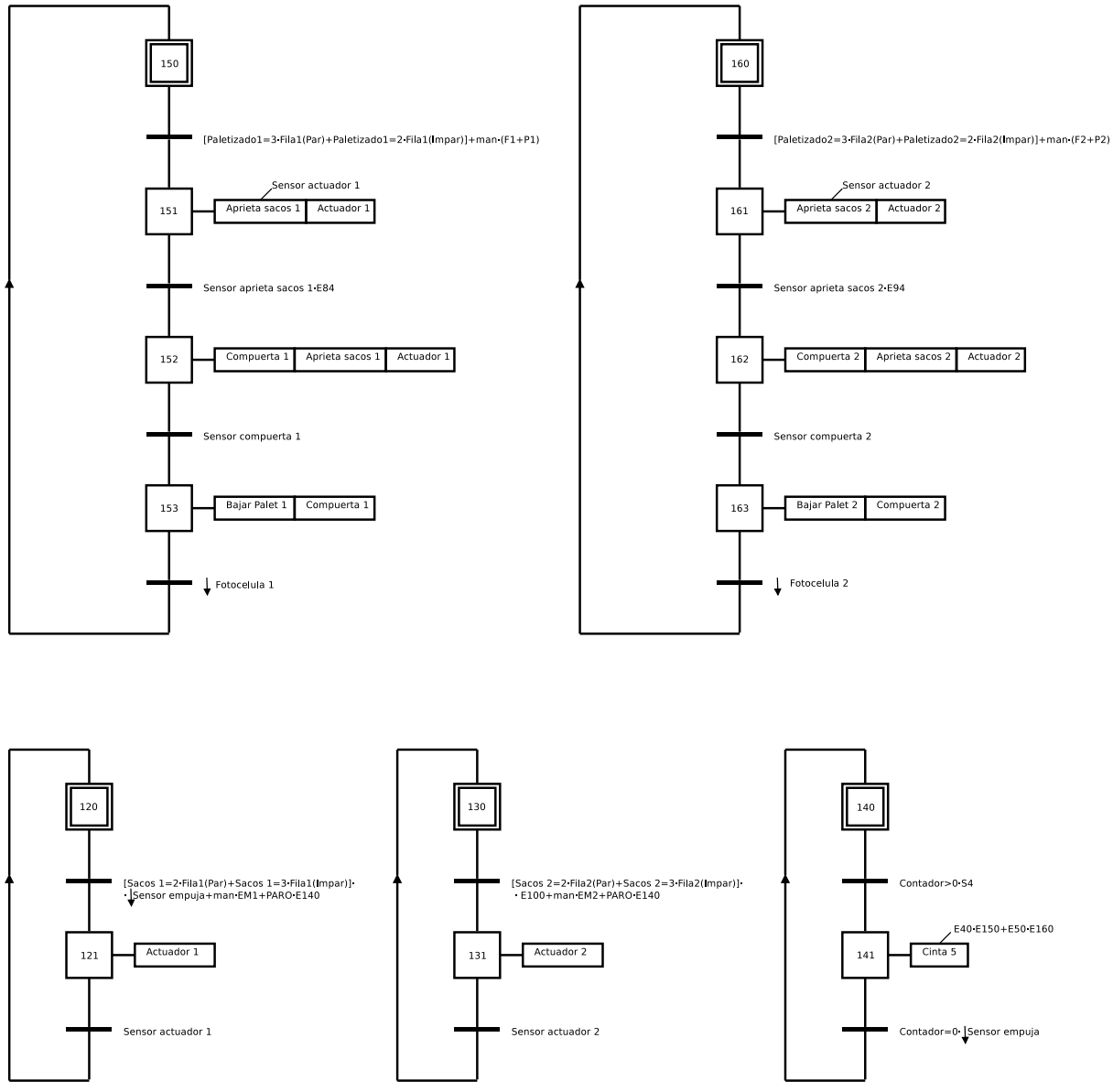
Programa de control

Diferentes Grafcet del sistema de control de fabricación de sacos y del sistema de paletizado automático.


0
(Marcha·PARO)·(auto+man·T1)
1
Cinta 1
Sensor Tolba 1
2
Trampilla 1
T/1/3seg·Sensor Plastico 1·E4
3
Sellador V1
Sellador H1
T/2/2seg
4
Cinta 3
10
(Marcha·PARO)·(auto+man·T2)
11
Cinta 2
Sensor Tolba 2
12
Trampilla 2
T/3/3seg·Sensor Plastico 2
13
E20·E15
14
Sellador V2
Sellador H2
T/4/2seg
15
Cinta 4
30
MAN
AUTO
31
32
MC+ALARMA
MC+ALARMA
50
Contador 2=6
51
E94
20
Sensor saco 1
21
Sensor saco 2
40
Contador 1=6
41
E84
60
Contador>0
61
Cinta 4 + S3
Cinta 3
Contador=0 · ↓S3
70
Contador>0·S3
71
Cinta 5 + S4
Cinta 4
Contador=0 · ↓S4
R
SETA
E31+E32
MP
SETA
E84+E94
PN
SETA
BotonParo
MC
SETA
E0·E10·E20·E30·E40·E50·E60·E70·E80·E90·E100·E110·E120·E130·E140·150·160
AL
SETA
ESPERA
RECUPERACION


80
Sensor palet 1·MARCHA
81
Cinta 6
Sensor palet 11
82
Subir Palet 1
Fotocelula 1
83
Bajar Palet 1
Fotocelula 1
84
(Filas 1=9+man·P1+
PARO·E120·Contador=0)·E150
85
Bajar Palet 1
Sensor palet 11
86
Cinta 6
Sensor fin 1
90
Sensor palet 2·MARCHA
91
Cinta 7
Sensor palet 22
92
Subir Palet 2
Fotocelula 2
93
Bajar Palet 2
Fotocelula 2
94
(Filas 2=9+man·P2+
PARO·E130·Contador=0)·E160
95
Bajar Palet 2
Sensor palet 22
96
Cinta 7
Sensor fin 2
100
auto·Sensor 2
101
man·EM5
Sensor empuja
102
Actuador 5
Sensor actuador 5


110
Sensor 1
Sensor 2
111
113
Sensor giro ·
Fila1(Par)
Sensor giro·
Fila1(Impar)
Sensor giro·
Fila2(Impar)
Sensor giro·
Fila2(Par)
112
Gira sacos
Sensor gira sacos


150
[Paletizado1=3·Fila1(Par)+Paletizado1=2·Fila1(Impar)]+man·(F1+P1)
Sensor actuador 1
151
Aprieta sacos 1
Actuador 1
Sensor aprieta sacos 1·E84
152
Compuerta 1
Aprieta sacos 1
Actuador 1
Sensor compuerta 1
153
Bajar Palet 1
Compuerta 1
Fotocelula 1
160
[Paletizado2=3·Fila2(Par)+Paletizado2=2·Fila2(Impar)]+man·(F2+P2)
Sensor actuador 2
161
Aprieta sacos 2
Actuador 2
Sensor aprieta sacos 2·E94
162
Compuerta 2
Aprieta sacos 2
Actuador 2
Sensor compuerta 2
163
Bajar Palet 2
Compuerta 2
Fotocelula 2
120
[Sacos 1=2·Fila1(Par)+Sacos 1=3·Fila1(Impar)]·
·Sensor empuja+man·EM1+PARO·E140
121
Actuador 1
Sensor actuador 1
130
[Sacos 2=2·Fila2(Par)+Sacos 2=3·Fila2(Impar)]·
·E100+man·EM2+PARO·E140
131
Actuador 2
Sensor actuador 2
140
Contador>0·S4
E40·E150+E50·E160
141
Cinta 5
Contador=0·Sensor empuja